


Steps:

1. Create the ec2 using the Ubuntu Ami


**Ubuntu Server 20.04 LTS (HVM), SSD Volume Type** - ami-09e67e426f25ce0d7 (64-bit x86) / ami-00d1ab6b335f217cf (64-bit Arm)
Free tier eligible
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes


Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

☒ 64-bit (x86)
☐ 64-bit (Arm)

Select

2. Select the subnet for the region you're in and enable it.

Network ⓘ
vpc-4231ae3f (default)  [Create new VPC](#)

Subnet ⓘ
subnet-a7f8b186 | Default in us-east-1a  [Create new subnet](#)
4088 IP Addresses available

Auto-assign Public IP ⓘ
Enable

Placement group ⓘ
☐ Add instance to placement group




3. Give your ec2 a name.

[Add another tag](#) (Up to 50 tags maximum)

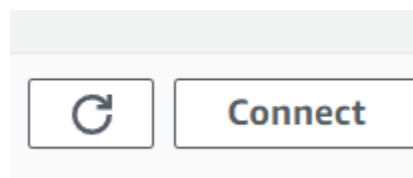
4. Set the security group rules as shown.

Security group name:		Docker			
Description:		launch-wizard-8 created 2021-10-14T00:29:26.359-04:00			
Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ	Description ⓘ	
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop	✕
Custom TCP	TCP	8080	Custom 0.0.0.0/0, :::0	e.g. SSH for Admin Desktop	✕

5. Once the ec2 is made and running, click the instanceID.

	Docker-agent	i-0ae912fbe6b06a2ae	 Running		t2.micro
---	--------------	---------------------	--	---	----------

6. Then click on connect.



7. Ensure that you're in the SSH client section then copy the address.

Connect to instance [Info](#)

Connect to your instance i-0ae912f6e6b06a2ae (Docker-agent) using any of these options

EC2 Instance Connect

Session Manager

SSH client

EC2 Serial Console

Instance ID

 i-0ae912f6e6b06a2ae (Docker-agent)

1. Open an SSH client.

2. Locate your private key file. The key used to launch this instance is EC2 Tutorial.pem


3. Run this command, if necessary, to ensure your key is not publicly viewable.


 `chmod 400 EC2 Tutorial.pem`

4. Connect to your instance using its Public DNS:

 `ec2-107-23-38-31.compute-1.amazonaws.com`

Example:

 `ssh -i "EC2 Tutorial.pem" ubuntu@ec2-107-23-38-31.compute-1.amazonaws.com`

 **Note:** In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

8. Once copied head inside your terminal to ssh into your instance. Ensure that your "pem" is typed correctly or you'll get an error.

```
ssh -i "EC2Tutorial.pem" ubuntu@ec2-107-23-38-31.compute-1.amazonaws.com
```

9. Once successful, follow the steps to in docker onto ubuntu from this site <https://serverspace.io/support/help/how-to-install-docker-on-ubuntu-20-04/>.

NOTE: ensure that you use the required username as shown below:

```
sudo usermod -aG docker ubuntu
```

10. Once you're done, use the command (`$ sudo docker images`) to test if docker was successfully installed.
11. Then use the command (`$ curl "https://awscli.amazonaws.com/awscli-exe-linux-86_64.zip" -o "aws.cliv2.zip"`).
12. Once the zip is downloaded use the command (`$ sudo apt-get install unzip`).
13. Then use the command (`$ sudo unzip aws.cliv2.zip`).
14. Then use the commands (`$ sudo ./aws/install`) and (`$ aws --version`) to see the version.
15. Once aws is successfully installed used the command (`$ aws configure`).
16. Fill in the information correctly then use the commands (`$ sudo apt update`) followed by (`$ sudo apt install default-jdk`).
17. Then use the command (`$ docker pull Jenkins/Jenkins`).
18. Then head into aws online and search for the Elastic Container Service(ECS). Once there create your Elastic Container Repository(ECR). Set the visibility to public and give it a name and keep the recommended settings.

ELK > Repositories > Create repository

Create repository

General settings

Visibility settings [Info](#)
Choose the visibility setting for the repository.

☐ Private
Access is managed by IAM and repository policy permissions.

☒ Public
Publicly visible and accessible for image pulls.

Once a repository is created, the visibility setting of the repository can't be changed.

Detail

Repository name [Info](#)
A namespace can be included with your repository name (e.g. namespace/repo-name).

public.ecr.aws/o1d2c8q0/

7 out of 205 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, and forward slashes.

19. Click your new repository and clicked push commands.



20. From push commands follow steps 1, 3 and 4. Ensure that you get a successful login from step 1 before moving on. After step 3 you can use the command (\$ docker images) to see the new repository made.

Push commands for jenkins

- Retrieve an authentication token and authenticate your Docker client to your registry.
Use the AWS CLI:

```
aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --password
```


Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
- Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions [here](#). You can skip this step if your image is already built:

```
docker build -t jenkins .
```
- After the build completes, tag your image so you can push the image to this repository:

```
docker tag jenkins:latest public.ecr.aws/o1d2c8q0/jenkins:latest
```
- Run the following command to push this image to your newly created AWS repository:

```
docker push public.ecr.aws/o1d2c8q0/jenkins:latest
```

Close

21. Once done, create a new cluster.

☒ New ECS Experience
[Tell us what you think](#)

Amazon ECS

Clusters

Task Definitions

Account Settings

Clusters

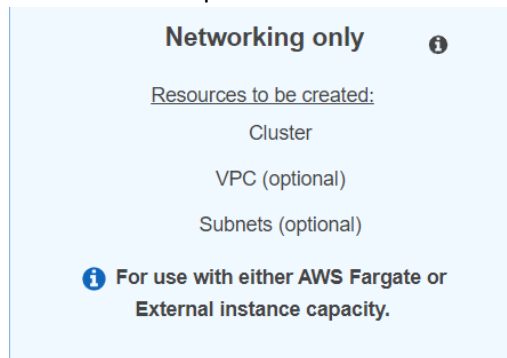
An Amazon ECS cluster is a regional grouping of EC2 instances that you manage using the Amazon ECS service. Clusters may contain more than one Amazon ECS service.

For more information, see the [ECS documentation](#).

Create Cluster

Get Started

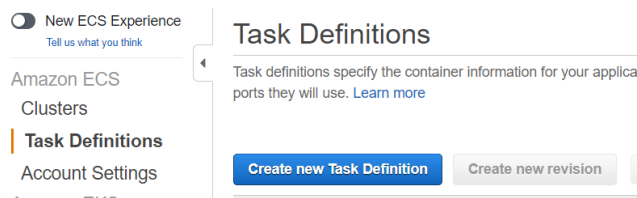
22. Select the first option



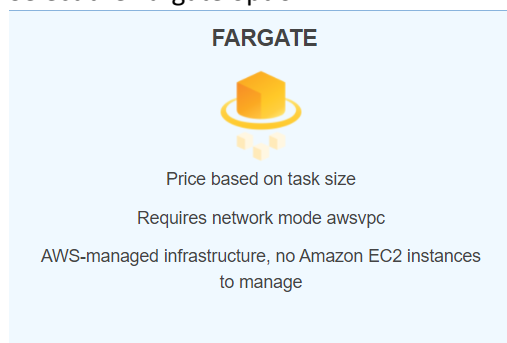
23. Then name your cluster and save changes.



24. Then create a task definition.



25. Select the Fargate option.



26. Name your task definition.



27. Most importantly ensure that you select a reasonable task size. This is important because you'll be using Jenkins and if it's too small the site is going to be extremely slow.

NOTE: fargate charges by the second so ensure that you are very careful and delete once done.

Task size

The task size allows you to specify a fixed size for your task. Task size is required for tasks using the Fargate launch type for the EC2 or External launch type. Container level memory settings are optional when task size is set. Task size is required for Windows containers.

Task memory (GB)

The valid memory range for 2 vCPU is: 4GB - 16GB.

Task CPU (vCPU)

The valid CPU range for 4GB memory is: 0.5 vCPU - 2 vCPU.

28. Add a new container



29. Give the container a name and copy the image made from the repository and paste it.

☐ jenkins public.ecr.aws/o1d2c8q0/jenkins

▼ Standard

Container name*

Image*

30. Set port mappings to 8080.

Port mappings

Container port	Protocol
<input type="text" value="8080"/>	<input type="text" value="tcp"/>

31. Go to the cluster you created and select Tasks.

Cluster : jenkins

Get a detailed view of the resources on your cluster.

Cluster ARN arn:aws:ecs:us-east-1:069598533

Status **ACTIVE**

Registered container instances 0

Pending tasks count 0 Fargate, 0 EC2, 0 External

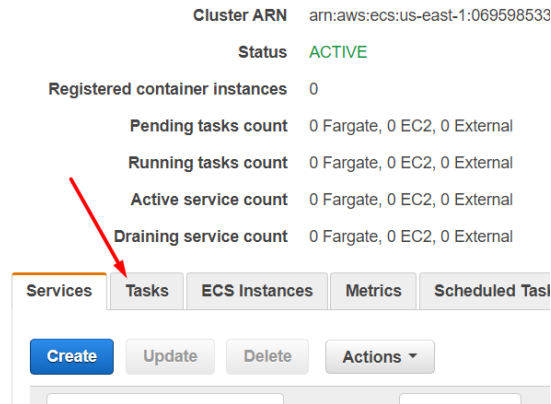
Running tasks count 0 Fargate, 0 EC2, 0 External

Active service count 0 Fargate, 0 EC2, 0 External

Draining service count 0 Fargate, 0 EC2, 0 External

Services **Tasks** ECS Instances Metrics Scheduled Tasl

Create Update Delete Actions ▾

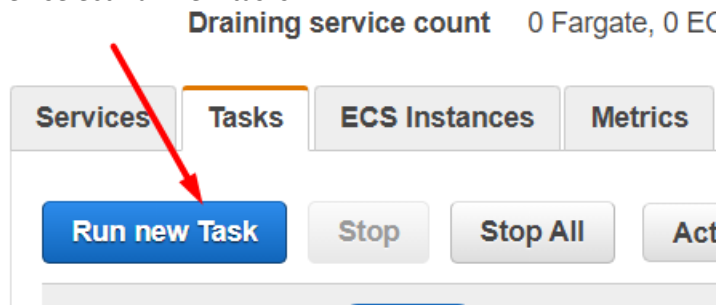


32. Then select Run new tasks.

Draining service count 0 Fargate, 0 EC

Services **Tasks** ECS Instances Metrics

Run new Task Stop Stop All Act



33. Select Fargate

Run Task

Select the cluster to run your task definition on and the number of copies of the target particular container instances, click Advanced Options.

Launch type ☒ FARGATE ☐ EC2 ☐ EXTERNAL

[Switch to capacity provider strategy](#)

34. Choose a vpc (I used the default) and choose a subnet (I used the one with the region I'm in.

VPC and security groups

VPC and security groups are configurable when your task definition uses the aws

Cluster VPC* vpc-4231ae3f (172.31.0.0/16) ⓘ

Subnets* subnet-a7f8b186
(172.31.80.0/20) - us-east-1a
assign ipv6 on creation: Disabled ⓘ

▼

35. Edit the security group as shown:

Configure security groups

A security group is a set of firewall rules that control the traffic for your task. On this page, you can add rules to allow specific traffic to reach your task or you can choose to use an existing security group. [Learn more.](#)

Assigned security groups ☒ Create new security group ☐ Select existing security group

Security group name* jenkins-377 ⓘ

Description Tue Oct 12 2021 13:38:26 GMT-0400 (Boli ⓘ

Inbound rules for security group

Type	Protocol	Port range	Source	
Custom TCP ▼	TCP	8080	Anywhere ▼	0.0.0.0/0, ::/0 ⓘ

[Add rule](#)

36. Save changes and wait for the task status to change from “pending” to “running”. Then copy and paste the Public IP in a new tab.

Network

Network mode awsvpc

ENI Id eni-08fcf63cf6a9c5177

Subnet Id subnet-a7f8b186

Private IP 172.31.81.118

Public IP 18.212.5.52

Mac address 12:1c:8c:5d:a1:11

Containers

Name	Container Runtime I...	Status ...	Image	Image Digi
jenkins-co...	ed5fcc7f34446ec984...	RUNNING	public.ecr.aws/o1d2c8q0/jenkins	

37. Once successful Jenkins will load on the new tab.



38. Once Jenkins loads successfully, head to the task and click on logs.

Task : ed5fcc7f334446ec984adb9045bf8ee3

Details Tags Logs

Cluster [jenkins](#)

Launch type FARGATE

Platform version 1.4.0

Task definition [jenkins-task:1](#)

Group family:jenkins-task

Task role None

Last status **RUNNING**

Desired status RUNNING

Created at 2021-10-12 13:41:57 -0400

Started at 2021-10-12 13:42:48 -0400

Network

Network mode awsvpc

ENI Id [eni-08fc63cf8a9c5177](#)

Subnet Id subnet-a7f6b186

Private IP 172.31.81.118

Public IP 18.212.5.52

39. Once there, you'll get the password required to log into Jenkins.

Task : ed5fcc7f334446ec984adb9045bf8ee3

Run more like this Stop

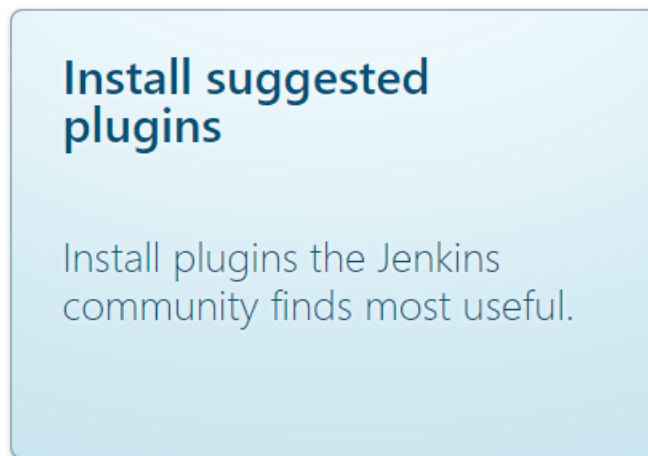
Details Tags Logs

Last updated on October 12, 2021 1:45:45 PM (0m ago)

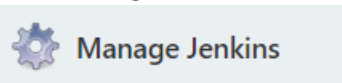
Filter logs

Timestamp (UTC+00:00)	Message
2021-10-12 13:43:29	2021-10-12 17:43:29.165+0000 [jd=43] INFO hudson.util.Retrier#start: Performed the action check updates server successfully at the attempt #1
2021-10-12 13:43:29	2021-10-12 17:43:29.049+0000 [jd=22] INFO hudson.WebAppMain\$3run: Jenkins is fully up and running
2021-10-12 13:43:29	2021-10-12 17:43:29.027+0000 [jd=28] INFO jenkins.initReactorRunner\$1onAttnained: Completed initialization
2021-10-12 13:42:58	2021-10-12 17:42:58.282+0000 [jd=29] INFO jenkins.install.SetupWizard#init:
2021-10-12 13:42:58	*****
2021-10-12 13:42:58	*****
2021-10-12 13:42:58	Jenkins initial setup is required. An admin user has been created and a password generated.
2021-10-12 13:42:58	Please use the following password to proceed to installation:
2021-10-12 13:42:58	47d424cc19d54ca7a5faa1b8aaac1cca
2021-10-12 13:42:58	This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

40. Install suggested plugins.



41. Click Manage Jenkins



42. Install both Docker pipeline and Amazon EC2 plugins.

Dashboard ▸ Plugin Manager

Updates

Available

Installed

Advanced

Install ↑	Name	Version	Released
<input checked="" type="checkbox"/>	<div><div>Docker Pipeline</div><div>Deployment DevOps docker pipeline</div><div>Build and use Docker containers from pipelines.</div></div>	1.26	7 mo 20 days ago
<input checked="" type="checkbox"/>	<div><div>Amazon EC2</div><div>agent aws Cloud Providers Cluster Management</div><div>This plugin integrates Jenkins with Amazon EC2 or anything implementing the EC2 API's such as an Ubuntu.</div></div>	1.65	7 days 1 hr ago

✓ Success

Amazon EC2

Restarting Jenkins

🔄 Pending

🔄 Pending

🔄 Pending

[Go back to the top page](#)
(you can start using the installed plugins right away)

☒ Restart Jenkins when installation is complete and no jobs are running

✓ Success

Amazon EC2

Restarting Jenkins

🔄 Pending

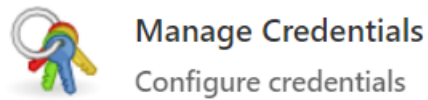
🔄 Pending

🔄 Pending

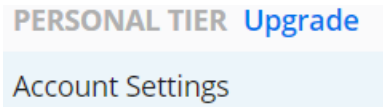
➡ [Go back to the top page](#)
(you can start using the installed plugins right away)

➡ ☒ Restart Jenkins when installation is complete and no jobs are running

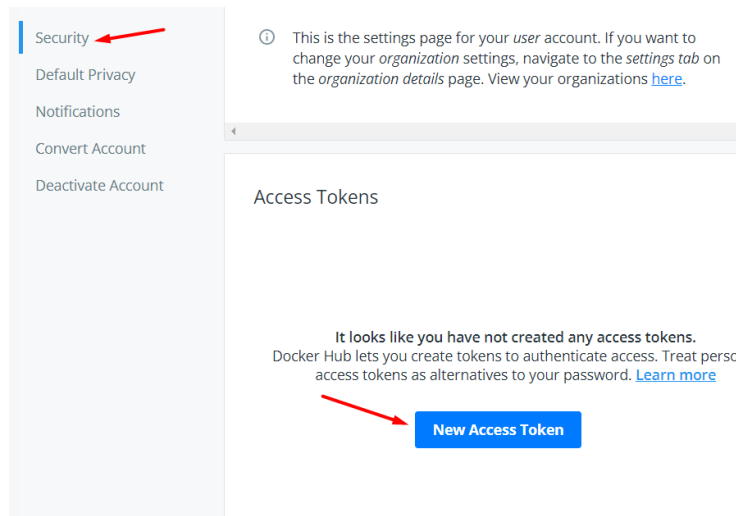
43. Then go to Manage Credentials.



44. When creating your docker credentials got to account settings.



45. Then go to security, there you will create a new access token if you don't already have one.



46. Then give your token a name and save it. Ensure that you copy the token and save it for future use.

47. Ensure that you choose the correct kind. Then use the username of your dockerhub account and use the access token for password. Give it an id and save it. Set credentials as shown

Kind

Username with password

Scope

Global (Jenkins, nodes, items, all child items, etc)

Username

mastercle

☐ Treat username as secret

Password

.....

ID

docker-id

48. Then set your Jenkins credentials as shown:

Kind

SSH Username with private key

Scope

Global (Jenkins, nodes, items, all child items, etc)

ID

ssh-jenkins-id

Description

Username

ubuntu

☐ Treat username as secret

Private Key

☒ Enter directly

Private Key

☒ Enter directly

Key

Enter New Secret Below

```
YfPJ+wK8gQCY/ws39qGMHTVHRWKZamcWVhUq1wbKb1HTqA9Cpy12Yfwr7ApA24sy
37T9dB4DTshkRH8HNFVwrEdNlxqpyw8KEZN9xfWK/TQLWEId81W5t82Bcs0i6BN5
DfQvhZvFrTH0wStBKslcXFDd8mKSnkZFpW2WiStm8zGeJ+WmqKpYg==
-----END RSA PRIVATE KEY-----
```

Passphrase

OK

NOTE: for private key, copy and paste your rsa private key.

49. Then add your credentials for github ensuring that the username is the same as your github account and password is your personal access token. Save changes after giving it an id.

Kind

Username with password

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

Username ?
Mastercle

☐ Treat username as secret ?

Password ?
.....

ID ?
github-id

Description ?

50. Once the credentials are set, time to add a new node.



Manage Nodes and Clouds

Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Node name

jenkins-agent

☒ **Permanent Agent**

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

OK

51. Configure your node accordingly giving it a name, correct number of executors (2), root directory, and a label name of your choice. For credentials, use your Jenkins credentials you created.

Name

jenkins-agent

Description

Number of executors

2

Remote root directory

/home/ubuntu/jenkins

Labels

jenkins-agent

Usage

Use this node as much as possible

Launch method

Launch agents via SSH

Host

172.31.88.49

Launch method

Launch agents via SSH

Host

172.31.88.49

Credentials

ubuntu

Add

Host Key Verification Strategy

Non verifying Verification Strategy

Availability

Keep this agent online as much as possible

Node Properties

☐ Disable deferred wipeout on this node

☐ Environment variables

☐ Tool Locations

Save

52. Ensure that your docker file is correct and your Jenkinsfile as well.

main DEPLOY07_ECS / Dockerfile

Mastercle Add files via upload

1 contributor

5 lines (3 sloc) | 90 Bytes

```
1 FROM openjdk:11
2
3 COPY ./demo-0.0.1-SNAPSHOT.jar app.jar
4
5 CMD ["java", "-jar", "app.jar"]
```



Mastercle Create Jenkinsfile

1 contributor

31 lines (28 sloc) | 736 Bytes

```
1 pipeline {
2     agent {
3         label "jenkins-agent"
4     }
5
6     environment {
7         DOCKERHUB_CREDENTIALS = credentials("docker-id")
8     }
9     stages {
10         stage('Build') {
11             steps {
12                 sh 'docker build -t mastercle/javadem .'
13                 sh 'echo "completed build"'
14             }
15         }
16
17         stage('Login') {
18             steps {
19                 sh 'echo $DOCKERHUB_CREDENTIALS_PSW | docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
20                 sh 'echo "completed login"'
21             }
22         }
23
24         stage('Push'){
25             steps {
26                 sh 'docker push mastercle/javadem:latest'
27                 sh 'echo "completed push"'
28             }
29         }
30     }
31 }
```

53. Then go to your dashboard to create a new item. Give it a name and select pipeline.

Enter an item name

» Required field



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

54. Set the pipeline in your Jenkins as shown, directing it to your github containing your Jenkins file. Use the github credentials you made earlier.

Pipeline

Definition

Pipeline script from SCM

SCM

Git

Repositories

Repository URL

https://github.com/Mastercle/DEPLOY07_ECS

Credentials

Mastercle/***** Add

Advanced...

Add Repository

Save Apply

Branches to build

Branch Specifier (blank for 'any')

*/main

Add Branch

Repository browser

(Auto)

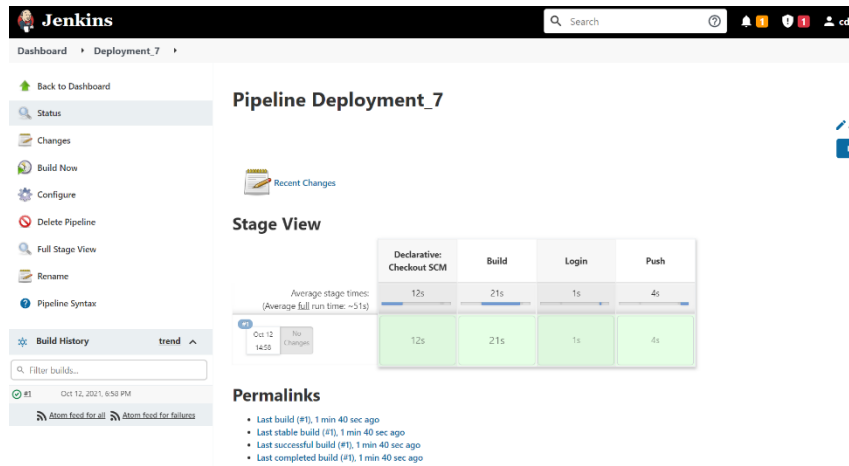
Additional Behaviours

Add

Script Path

Jenkinsfile

55. Save changes and build your pipeline.

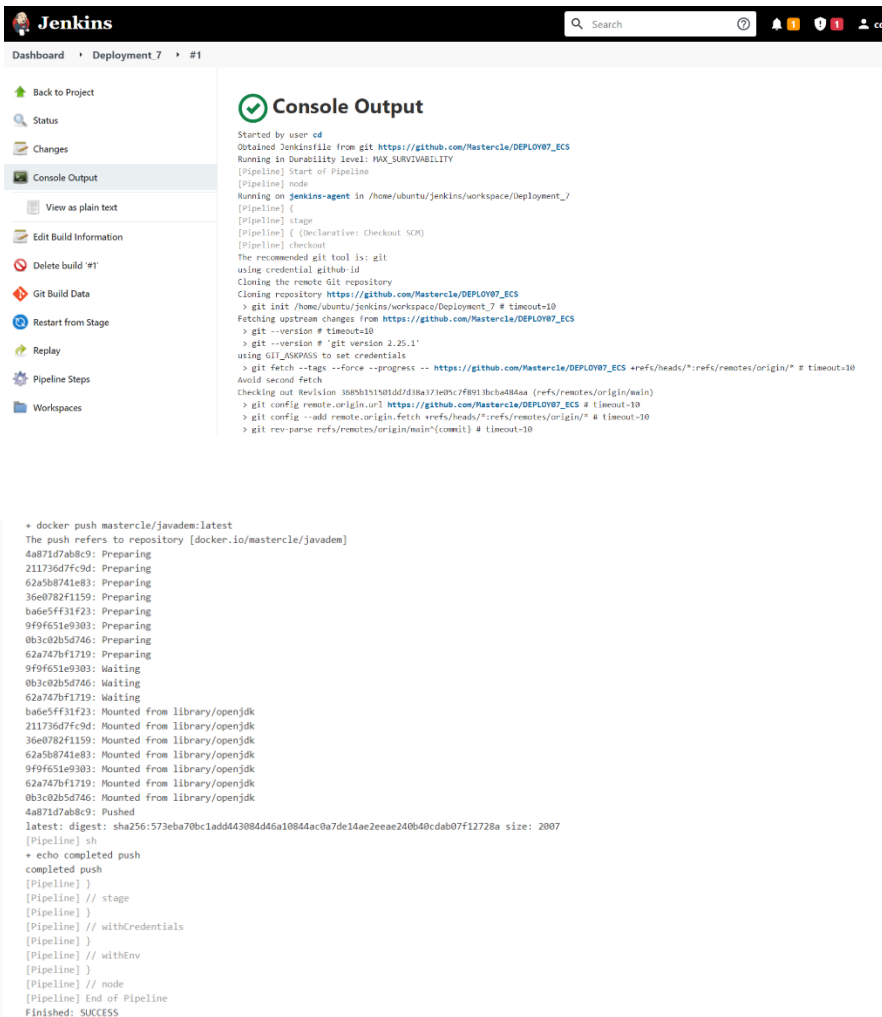


The Jenkins dashboard for Pipeline Deployment_7 shows a successful build. The left sidebar contains navigation links: Back to Dashboard, Status, Changes, Build Now, Configure, Delete Pipeline, Full Stage View, Rename, Pipeline Syntax, Build History (selected), and trend. The main area displays the Pipeline View with a table of stage times:

	Declarative: Checkout SCM	Build	Login	Push
Average stage times (Average full run time: ~51s)	12s	21s	1s	4s
Oct 12 14:58	12s	21s	1s	4s

Below the table, the Permalinks section lists:

- Last build (#1), 1 min 40 sec ago
- Last stable build (#1), 1 min 40 sec ago
- Last successful build (#1), 1 min 40 sec ago
- Last completed build (#1), 1 min 40 sec ago




The Jenkins console output for Pipeline Deployment_7 shows the following steps:

```
Started by user cd
Obtained Jenkinsfile from git https://github.com/Mastercle/DEPLOY07_ECS
Running in Durability Level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on jenkins-agent in /home/ubuntu/jenkins/workspace/Deployment_7
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
The recommended git tool is: git
using credential github-id
Cloning the remote git repository
Cloning repository https://github.com/Mastercle/DEPLOY07_ECS
> git init /home/ubuntu/jenkins/workspace/Deployment_7 # timeout=10
Fetching upstream changes from https://github.com/Mastercle/DEPLOY07_ECS
> git --version # timeout=10
> git --version # 'git version 2.25.1'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/Mastercle/DEPLOY07_ECS +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Checking out Revision 3a8b515101d7d18a37b09c7f8013bca38aa (refs/remotes/origin/main)
> git config remote.origin.url https://github.com/Mastercle/DEPLOY07_ECS # timeout=10
> git config --add remote.origin.fetch refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10

* docker push mastercle/javadem:latest
The push refers to repository [docker.io/mastercle/javadem]
4a871d7ab8c9: Preparing
211736dfc9d: Preparing
62a5b8741e83: Preparing
36e0782f1159: Preparing
ba6e5ff31f23: Preparing
9f9f651e9303: Preparing
0b3c02b5d746: Preparing
62a747bf1719: Preparing
9f9f651e9303: Waiting
0b3c02b5d746: Waiting
62a747bf1719: Waiting
ba6e5ff31f23: Mounted from library/openjdk
211736dfc9d: Mounted from library/openjdk
36e0782f1159: Mounted from library/openjdk
62a5b8741e83: Mounted from library/openjdk
9f9f651e9303: Mounted from library/openjdk
62a747bf1719: Mounted from library/openjdk
0b3c02b5d746: Mounted from library/openjdk
4a871d7ab8c9: Pushed
latest: digest: sha256:573eba70bc1add443084d46a10844ac0a7de14ac2eeae240d40cda07f12728a size: 2007
[Pipeline] sh
* echo completed push
completed push
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

56. Once it's successful head into your dockerhub to see the image made.

 Search for great content

Explore Repositories Organizations Help ▾

Upgrade

mastercle | ▾

Search by repository name

Create Repository


mastercle / **javadem**
Updated 7 minutes ago


Not Scanned


☆ 0

↓ 4

Public

 mastercle / **javadem**

This repository does not have a description 

 Last pushed: 8 minutes ago

Public View

Docker commands

To push a new tag to this repository,

```
docker push mastercle/javadem:tagname
```

Tags and Scans

VULNERABILITY SCANNING - DISABLED
[Enable](#)

This repository contains 1 tag(s).