

## 1. Create a customized Docker image using a Dockerfile.

- mkdir custom-image
- cd custom-image

```
root@ip-172-31-119-6:~/custom-image# mkdir custom-image
root@ip-172-31-119-6:~/custom-image# cd custom-image
```

To create and store data in a file.

- echo "<h1>Hello from my custom Docker image!</h1>"> index.html

vi Dockerfile and add this script

```
# Use official nginx base image
FROM nginx:latest

# Copy custom HTML to nginx web directory
COPY index.html /usr/share/nginx/html/index.html

# Expose port
EXPOSE 80

# Start Nginx server
CMD ["nginx", "-g", "daemon off;"]
```

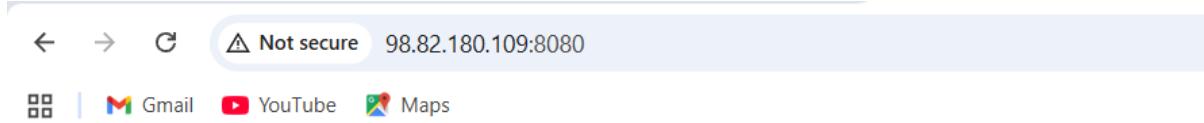
- docker build -t my-custom-image:1.0 .

```
[root@ip-172-31-119-6 custom-image]# ls
Dockerfile index.html
[root@ip-172-31-119-6 custom-image]# docker build -t my-custom-image:1.0 .
[+] Building 4.8s (7/7) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 324B
=> [internal] load metadata for docker.io/library/nginx:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build context
=> => transferring context: 141B
=> [1/2] FROM docker.io/library/nginx@sha256:553f64aecdc31b5bf944521731cd70e35da4f
=> => resolve docker.io/library/nginx@sha256:553f64aecdc31b5bf944521731cd70e35da4f
=> => sha256:53d743880af45adf9f141eec1fe3a413087e528075a5d8884d6215ddfdd2b806 954B / 954B
=> => sha256:5c733364e9a8f7e6d7289ceaad623c6600479fe95c3ab5534f07bfd7416d9541 2.29kB / 2.
=> => sha256:0e4bc2bd6656e6e004e3c749af70e5650bac2258243eb0949dea51cb8b7863db 29.78MB / 2
=> => sha256:b5feb73171bf1bcf29fdd1ba642c3d30cdf4c6329b19d89be14d209d778c89ba 29.97MB / 2
=> => sha256:553f64aecdc31b5bf944521731cd70e35da4faed96b2b7548a3d8e2598c52a42 10.23kB / 1
=> => sha256:60adc2e137e757418d4d771822fa3b3f5d3b4ad58ef2385d200c9ee78375b6d5 8.75kB / 8.
=> => sha256:108ab82928207abd9abfd960dd842364037563fc560b8f6304e4a91454fe 628B / 628B
=> => sha256:77fa2eb0631772679b0e48eca04f4906fba5fe94377e01618873a4a1171107ce 404B / 404B
=> => sha256:192e2451f8751fb74549c932e26a9bcfd7b669fe2f5bd8381ea5ac65f09b256b 1.21kB / 1.
=> => sha256:de57a609c9d5148f10b38f5c920d276e9e38b2856fe16c0aae1450613dc12051 1.40kB / 1.
=> => extracting sha256:0e4bc2bd6656e6e004e3c749af70e5650bac2258243eb0949dea51cb8b7863db
=> => extracting sha256:b5feb73171bf1bcf29fdd1ba642c3d30cdf4c6329b19d89be14d209d778c89ba
=> => extracting sha256:108ab82928207abd9abfd960dd842364037563fc560b8f6304e4a91454fe
=> => extracting sha256:53d743880af45adf9f141eec1fe3a413087e528075a5d8884d6215ddfdd2b806
=> => extracting sha256:77fa2eb0631772679b0e48eca04f4906fba5fe94377e01618873a4a1171107ce
=> => extracting sha256:192e2451f8751fb74549c932e26a9bcfd7b669fe2f5bd8381ea5ac65f09b256b
=> => extracting sha256:de57a609c9d5148f10b38f5c920d276e9e38b2856fe16c0aae1450613dc12051
```

- docker run -d -p 8080:80 my-custom-image:1.0

```
[root@ip-172-31-119-6 custom-image]# docker run -d -p 8080:80 my-custom-image:1.0
c03781b6fed481c0a51dbed4e6ec953cb49f6e755a94b7e5352a0f77210342bd
```

Go and search in the browser with the public\_ip and port number 8080.



## 2. Push the image to Docker Hub.

\* docker login

```
[root@ip-172-31-119-6 ~]# docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have a Docker ID, head over to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants better security and permissions using SSO. Learn more at https://docs.docker.com/go/access-tokens/
Username: mujaheed00
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
```

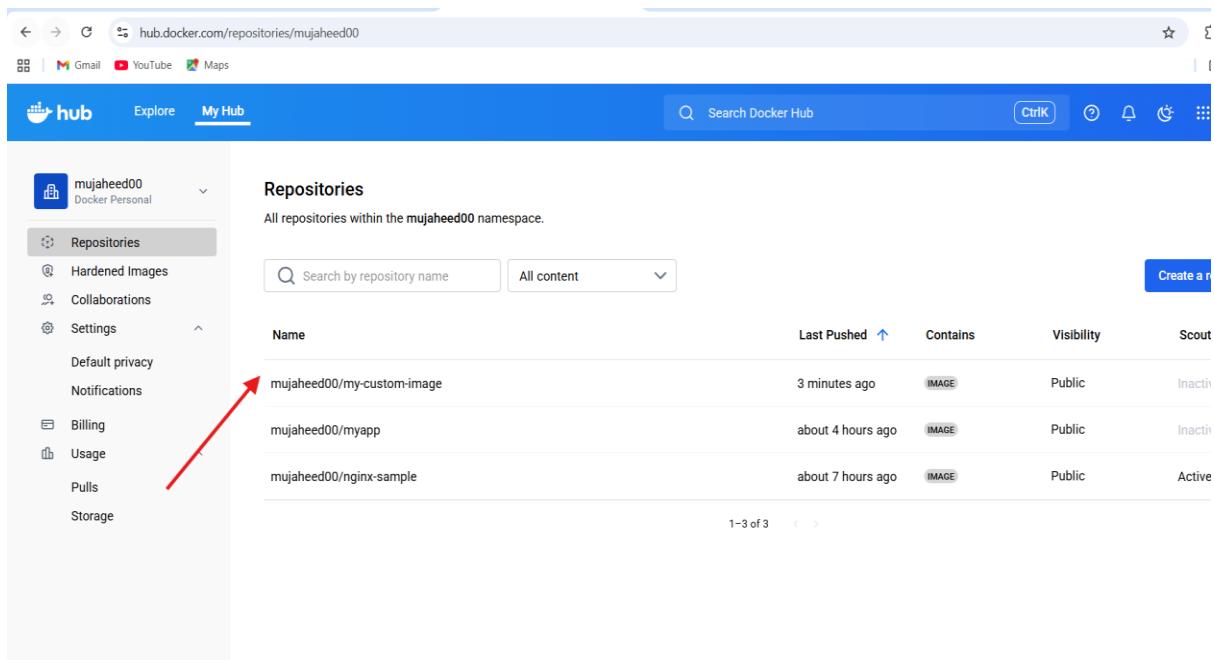
- docker tag my-custom-image:1.0 mujaheed00/my-custom-image:1.0

```
[root@ip-172-31-119-6 ~]# docker tag my-custom-image:1.0 mujaheed00/my-custom-image:1.0
```

- docker push mujaheed00/my-custom-image:1.0

```
[root@ip-172-31-119-6 ~]# docker push mujaheed00/my-custom-image:1.0
The push refers to repository [docker.io/mujaheed00/my-custom-image]
b3a1959b9926: Pushed
38d44e06fd01: Mounted from library/nginx
388bb4cadb9e: Mounted from library/nginx
5f0d4d15245b: Mounted from library/nginx
fe0771a36433: Mounted from library/nginx
1e79db1a7c1e: Mounted from library/nginx
008ba900efaf1: Mounted from library/nginx
70a290c5e58b: Mounted from library/nginx
1.0: digest: sha256:194ceccfdec8adbee1361276d2d1465fc59f0407a1c1f59da4a389b6a0936b0e size: 1985
[root@ip-172-31-119-6 ~]#
```

The image has been pushed to dockerhub.



The screenshot shows the Docker Hub 'My Hub' interface. On the left, there is a sidebar with the user's profile picture and name 'mujaheed00'. Below the profile, there are several menu items: 'Repositories' (which is currently selected and highlighted in grey), 'Hardened Images', 'Collaborations', 'Settings', 'Default privacy', 'Notifications', 'Billing', 'Usage', 'Pulls', and 'Storage'. The main content area is titled 'Repositories' and displays a table of repositories. The table has columns for 'Name', 'Last Pushed', 'Contains', 'Visibility', and 'Scout'. There are three entries in the table:

Name	Last Pushed	Contains	Visibility	Scout
mujaheed00/my-custom-image	3 minutes ago	IMAGE	Public	Inactive
mujaheed00/myapp	about 4 hours ago	IMAGE	Public	Inactive
mujaheed00/nginx-sample	about 7 hours ago	IMAGE	Public	Active

At the bottom of the table, it says '1–3 of 3'.

mujahed00/my-custom-image

Last pushed 3 minutes ago • ⭐0 • ⏺0

Add a description *(i)*

Add a category *(i)*

General Tags Image Management BETA Collaborators Webhooks Settings

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
1.0		Image	less than 1 day	3 minutes

See all

Repository overview (i) INCOMPLETE

An overview describes what your image does and how to run it. It displays in [the public view of your repository](#) once you have pushed some content.

Docker Build Cloud

Build with Docker Build Cloud

Accelerate image build and shared cache.

Docker Build Cloud enables infrastructure with Docker Build Cloud

Get faster builds through multi-platform support managing infrastructure.

Go to Docker Build Cloud

### 3. Push the same image to Amazon ECR.

- aws configure

```
[root@ip-172-31-119-6 ~]# aws configure
AWS Access Key ID [*****XKPJ]: AKIATNTADWL TUPIUCV7
AWS Secret Access Key [*****3n6u]: VXjvEq01gjoQmn1GE0JfgFbaGuqTl x0HXPbeqCqk
Default region name [us-east-1]: us-east-1
Default output format [json]: json
[root@ip-172-31-119-6 ~]# |
```

- aws ecr create-repository --repository-name my-custom-image

```
[root@ip-172-31-119-6 ~]# aws ecr create-repository --repository-name my-custom-image
{
  "repository": {
    "repositoryArn": "arn:aws:ecr:us-east-1:235351028455:repository/my-custom-image",
    "registryId": "235351028455",
    "repositoryName": "my-custom-image",
    "repositoryUri": "235351028455.dkr.ecr.us-east-1.amazonaws.com/my-custom-image",
    "createdAt": "2025-11-26T16:40:48.128000+00:00",
    "imageTagMutability": "MUTABLE",
    "imageScanningConfiguration": {
      "scanOnPush": false
    },
    "encryptionConfiguration": {
      "encryptionType": "AES256"
    }
  }
}
[root@ip-172-31-119-6 ~]#
```

- aws ecr get-login-password --region us-east-1 | \docker login --username AWS --password-stdin  
235351028455.dkr.ecr.us-east-1.amazonaws.com

```
[root@ip-172-31-119-6 ~]# aws ecr get-login-password --region us-east-1 | \
docker login --username AWS --password-stdin 235351028455.dkr.ecr.us-east-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

- docker tag my-custom-image:1.0  
23531028455.dkr.ecr.us-east-1.amazonaws.com/my-custom-image:1.0

```
[root@ip-172-31-119-6 ~]# docker tag my-custom-image:1.0 23531028455.dkr.ecr.us-east-1.amazonaws.com/my-custom-image:1.0
```

- docker push 235351028455.dkr.ecr.us-east-1.amazonaws.com/my-custom-image:1.0

```
[root@ip-172-31-119-6 ~]# docker push 235351028455.dkr.ecr.us-east-1.amazonaws.com/my-custom-image:1.0
The push refers to repository [235351028455.dkr.ecr.us-east-1.amazonaws.com/my-custom-image]
b3a1959b9926: Pushed
38d44e06fd01: Pushed
388bb4cadb9e: Pushed
5f0d4d15245b: Pushed
fe0771a36433: Pushed
1e79db1a7c1e: Pushed
008ba900efa1: Pushed
70a290c5e58b: Pushed
1.0: digest: sha256:1a14c1478781e500fdb5bf5d5aaa4304a8ff31ec1f847a06d630bc501d529d82 size: 1985
```

Go to ECR and select repositories.

The screenshot shows the AWS ECR console. On the left, there's a sidebar with navigation links for 'Amazon Elastic Container Registry', 'Private registry' (selected), 'Public registry', and 'ECR public gallery'. The main area is titled 'Private repositories (1)' and shows a table with one row. The row details a repository named 'my-custom-image' with the URI '235351028455.dkr.ecr.us-east-1.amazonaws.com/my-custom-image'. The table includes columns for 'Repository name', 'URI', 'Created at', 'Tag immutability', and 'Encryption type'. Buttons for 'View push commands', 'Delete', 'Actions', and 'Create repository' are at the top right.

A image ha been created.

The screenshot shows the 'Images' tab for the 'my-custom-image' repository. The sidebar on the left is identical to the previous screenshot. The main area shows a table of images. There is one image listed: '1.0' (Image type, created on November 26, 2025, 22:17:09 UTC+05.5, size 60.76 MB, digest sha256:1a14c147...). Buttons for 'Delete', 'Copy URI', 'Details', 'Scan', and 'View push commands' are at the top right of the image list.

## 4. Provision one EC2 instance using Terraform and install Jenkins.

Give the script in main.tf

```
resource "aws_instance" "my_ec2" {
    ami      = "ami-0fa3fe0fa7920f68e"
    instance_type = "t3.micro"
    key_name      = "red"
    subnet_id     = "subnet-0a192382de0e2bf6a"
```

```
vpc_security_group_ids = [  
    aws_security_group.ec2_sg.id  
]
```

```
tags = {  
    Name = "MyTerraformEC2"  
}  
}
```

```
resource "aws_security_group" "ec2_sg" {  
    name      = "ec2-basic-sg"  
    description = "Allow SSH"
```

```
    ingress {  
        description = "SSH"  
        from_port   = 22  
        to_port     = 22  
        protocol    = "tcp"  
        cidr_blocks = ["0.0.0.0/0"]  
    }
```

```

egress {

  from_port  = 0

  to_port    = 0

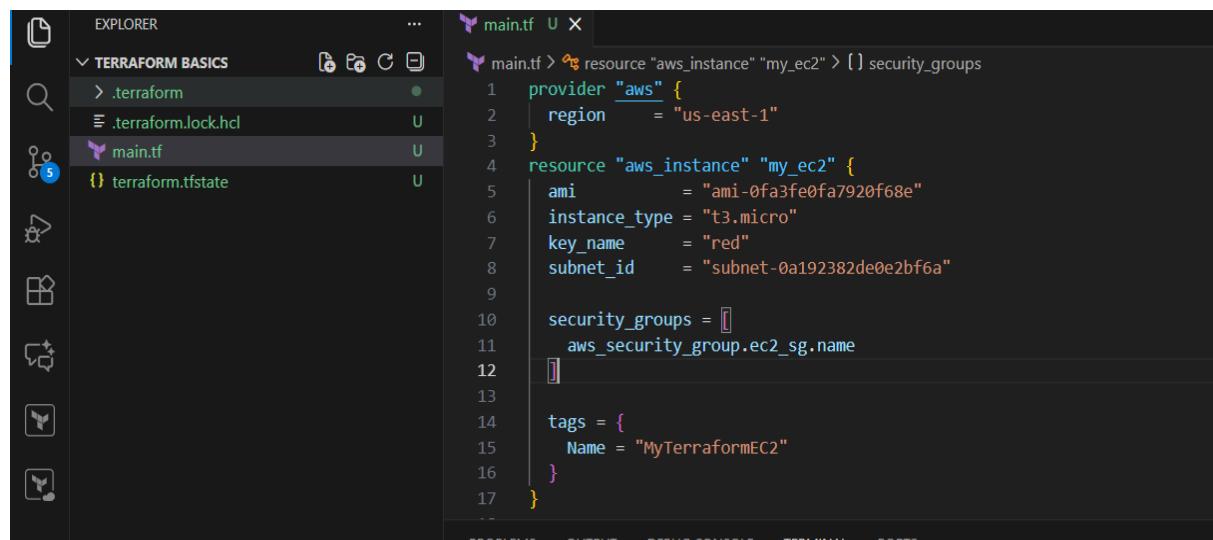
  protocol   = "-1"

  cidr_blocks = ["0.0.0.0/0"]

}

}

```



```

main.tf
1 provider "aws" {
2   region      = "us-east-1"
3 }
4 resource "aws_instance" "my_ec2" {
5   ami          = "ami-0fa3fe0fa7920f68e"
6   instance_type = "t3.micro"
7   key_name     = "red"
8   subnet_id    = "subnet-0a192382de0e2bf6a"
9
10  security_groups = [
11    aws_security_group.ec2_sg.name
12  ]
13
14  tags = {
15    Name = "MyTerraformEC2"
16  }
17

```

- terraform init
- terraform apply

```

PS C:\Users\Ashish\Desktop\Terraform basics> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.23.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\Ashish\Desktop\Terraform basics> terraform validate
Success! The configuration is valid.

```

```

}

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_instance.my_ec2: Creating...
aws_instance.my_ec2: Still creating... [00m10s elapsed]
aws_instance.my_ec2: Creation complete after 16s [id=i-005f8e5a4fb10e328]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics>

```

An instance has been created.

The screenshot shows the AWS Management Console with the EC2 service selected. In the left navigation pane, 'Instances' is expanded, showing 'Instances' and 'Launch Templates'. The main content area displays a table titled 'Instances (1/1)'. The table has one row for 'MyTerraformEC2', which is listed under the 'Name' column. The 'Instance ID' is 'i-005f8e5a4fb10e328'. The 'Instance state' is 'Running', indicated by a green circle with a checkmark. The 'Instance type' is 't3.micro', 'Status check' is 'Initializing', and it is located in 'us-east-1a'. There are buttons for 'Connect', 'Actions', and 'Launch instances' at the top of the table.

To install Jenkins give this script as userdata.

**user\_data = <<-EOF**

```
#!/bin/bash
```

```
yum update -y
yum install java-17-amazon-corretto -y
wget -O /etc/yum.repos.d/jenkins.repo
https://pkg.jenkins.io/redhat-stable/jenkins.repo
rpm --import https://pkg.jenkins.io/redhat-
stable/jenkins.io-2023.key
yum install -y jenkins
systemctl start jenkins
systemctl enable jenkins
EOF
```

And also add open port for Jenkins.

```
ingress {
    description = "Jenkins UI"
    from_port = 8080
    to_port = 8080
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
}
```

```

main.tf
1 resource "aws_instance" "my_ec2" {
  user_data = <<-EOF
  #!/bin/bash
  yum update -y
  # Install Java (required for Jenkins)
  yum install java-17-amazon-corretto -y
  # Add Jenkins repo
  wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
  rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
  #
  # Install Jenkins
  yum install -y jenkins
  # Start & enable Jenkins
}

```

```

41   from_port    = 22
42   to_port      = 22
43   protocol     = "tcp"
44   cidr_blocks = ["0.0.0.0/0"]
45 }
46 ingress {
47   description = "Jenkins UI"
48   from_port   = 8080
49   to_port     = 8080
50   protocol    = "tcp"
51   cidr_blocks = ["0.0.0.0/0"]
52 }
53

```

- `terraform init`
- `terraform apply`

```

PS C:\Users\Ashish\Desktop\Terraform basics> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.23.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\Ashish\Desktop\Terraform basics> terraform apply
aws_security_group.ec2_sg: Refreshing state... [id=sg-0de31d862e3786d38]
aws_instance.my_ec2: Refreshing state... [id=i-005f8e5a4fb10e328]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated
  ~ update in-place

Terraform will perform the following actions:

# aws_instance.my_ec2 will be updated in-place
~ resource "aws_instance" "my_ec2" {

```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

# (9 unchanged attributes hidden)
}

Plan: 0 to add, 2 to change, 0 to destroy.
Plan: 0 to add, 2 to change, 0 to destroy.

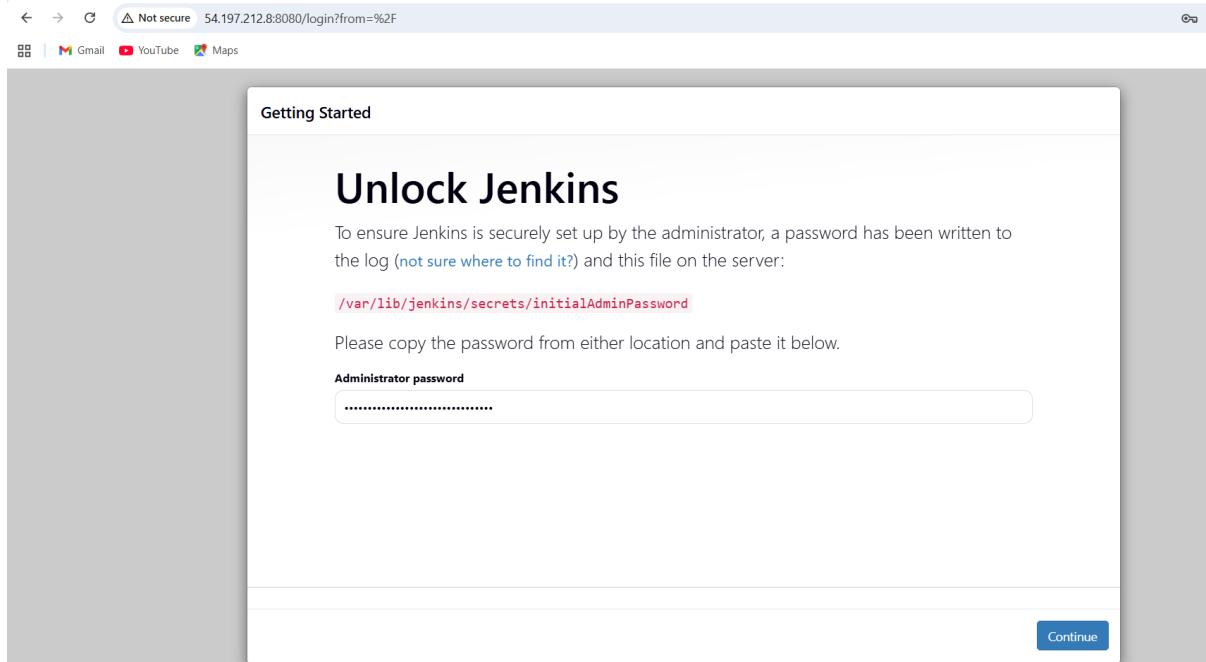
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

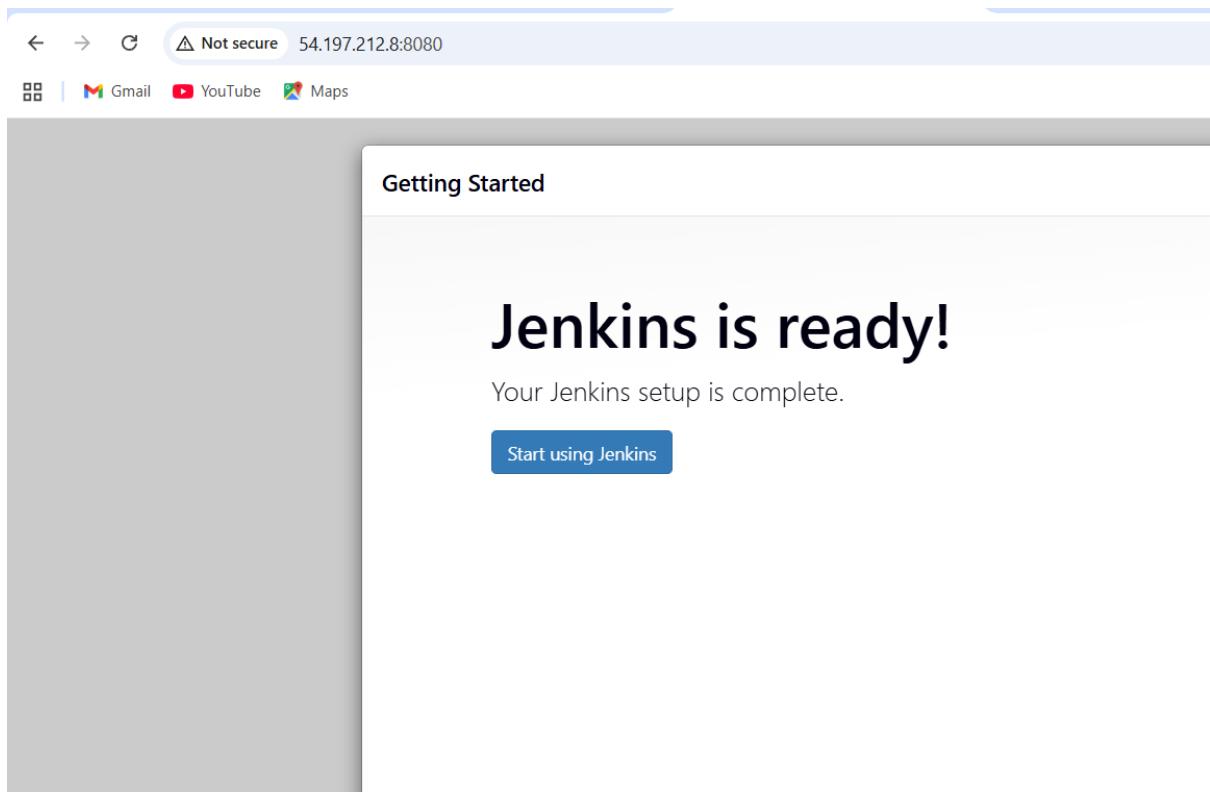
Enter a value: yes

aws_security_group.ec2_sg: Modifying... [id=sg-0de31d862e3786d38]
aws_security_group.ec2_sg: Modifications complete after 3s [id=sg-0de31d862e3786d38]
aws_instance.my_ec2: Modifying... [id=i-005f8e5a4fb10e328]
aws_instance.my_ec2: Still modifying... [id=i-005f8e5a4fb10e328, 00m10s elapsed]
aws_instance.my_ec2: Still modifying... [id=i-005f8e5a4fb10e328, 00m20s elapsed]
aws_instance.my_ec2: Still modifying... [id=i-005f8e5a4fb10e328, 00m30s elapsed]
aws_instance.my_ec2: Modifications complete after 37s [id=i-005f8e5a4fb10e328]

Apply complete! Resources: 0 added, 2 changed, 0 destroyed.
PS C:\Users\Ashish\Desktop\Terraform basics>
```

Search on browser by using public\_ip and port number 8080.





A screenshot of the Jenkins dashboard. The URL in the address bar is '54.197.212.8:8080'. The page title is 'Jenkins' with a logo. On the left, there are two cards: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (0/2). On the right, the main area is titled 'Welcome to Jenkins!' with the subtext 'This page is where your Jenkins jobs will be displayed. To get started, you can create new items or start building a software project.' It includes a 'Create a job' button and a 'Set up a distributed build' section with links for 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'.

## 5. Create one Jenkins job to build and push the Docker image to Docker Hub.

- Source: <https://github.com/betawins/Python-app.git>



Source Codes: <https://github.com/betawins/docker-tasks.git>

- From the frontend source code, write a Dockerfile, build a Docker image, run it, and push that image to your Docker registry.
- From the Java-based source code, write a Dockerfile, build, run, and push the image to the Docker registry.
- From the Node.js-based source code, write a Dockerfile, build with tag v1, run, and push it to the Docker registry.
- Write a docker-compose file to set up WordPress with a MySQL database.

Make sure you have your Jenkins server and docker server in the same instance.

- yum install docker -y
- systemctl start docker

```
[root@ip-172-31-109-235 ~]# yum install docker -y
Last metadata expiration check: 0:45:23 ago on Thu Nov 27 09:16:41
Dependencies resolved.
=====
 Package           Architecture      Version
=====
Installing:
 docker            x86_64          25.0.13-1.amzn2023.0.2.x86_64
=====
Transaction Summary
=====
Install 1 Package
=====
Total size: 46 M
Installed size: 179 M
Downloading Packages:
[SKIPPED] docker-25.0.13-1.amzn2023.0.2.x86_64.rpm: Already downloaded
Running transaction check
```

- usermod -aG docker Jenkins
- systemctl restart jenkins

```
[root@ip-172-31-109-235 ~]# usermod -aG docker jenkins
[root@ip-172-31-109-235 ~]# systemctl restart jenkins
```

Go to Jenkins, manage Jenkins,plugins,available plugins.Install

- Docker pipeline

The screenshot shows a browser window with the Jenkins plugin manager interface. The address bar indicates the URL is 54.162.209.181:8080/manage/pluginManager/installed. The page title is "Jenkins / Manage Jenkins / Plugins". On the left, there's a sidebar with links for "Updates", "Available plugins", "Installed plugins" (which is currently selected and highlighted in blue), and "Advanced settings". A search bar at the top right contains the text "docker Pi". Below the search bar, a table lists the "Docker Pipeline" plugin, showing its name, version (634.vedc7242b\_eda\_7), and a brief description: "Build and use Docker containers from pipelines." There is also a link to "Report an issue with this plugin".

Go to manage Jenkins,credentials,system,global credentials and give the docker hub username and password.

← → ⌛ Not secure 54.162.209.181:8080/manage/credentials/store/system/domain/\_/newCredentials

Gmail YouTube Maps

 Jenkins / Manage Jenkins / Credentials / System / Global credentials (unrestr... ▾

## New credentials

Kind

Username with password

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

Username ?  
mujaheed00

Treat username as secret ?

Password ?  
.....

ID ?  
dockerhub-creds

**Create**

Click on new item and name it as Build-and-Push-Docker and select type as pipeline.

← → ⌛ Not secure 54.162.209.181:8080/newJob

Gmail YouTube Maps

 Jenkins / New Item

## New Item

Enter an item name  
Build-and-Push-Docker

Select an item type

 Freestyle project  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by steps like archiving artifacts and sending email notifications.

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

 Multi-configuration project  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments or platform-specific builds, etc.

 Folder  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which just creates a separate namespace, so you can have multiple things of the same name as long as they are in folders.

**OK**

- cd /var/lib/jenkins/workspace/Build-and-Push-Docker
- mkdir docker-tasks
- cd docker-tasks

```
[root@ip-172-31-109-235 ~]# cd /var/lib/jenkins/workspace/
[root@ip-172-31-109-235 workspace]# cd /var/lib/jenkins/workspace/Build-and-Push-Docker/
[root@ip-172-31-109-235 Build-and-Push-Docker]# cd docker-tasks
[root@ip-172-31-109-235 Build-and-Push-Docker]# git clone https://github.com/betawins/docker-tasks.git
Cloning into 'docker-tasks'...
remote: Enumerating objects: 146, done.
remote: Counting objects: 100% (146/146), done.
remote: Compressing objects: 100% (123/123), done.
remote: Total 146 (delta 12), reused 141 (delta 9), pack-reused 0 (from 0)
Receiving objects: 100% (146/146), 22.11 MiB | 72.79 MiB/s, done.
Resolving deltas: 100% (12/12), done.
[root@ip-172-31-109-235 Build-and-Push-Docker]# ls
docker-tasks
[root@ip-172-31-109-235 Build-and-Push-Docker]# cd docker-tasks
[root@ip-172-31-109-235 docker-tasks]# ls
Frontend_based_source Java_based_source NodeJs_based_source README.md
[root@ip-172-31-109-235 docker-tasks]# mkdir frontend
mkdir java
mkdir node
[root@ip-172-31-109-235 docker-tasks]# mv Frontend_based_source frontend/
[root@ip-172-31-109-235 docker-tasks]# mv Java_based_source java/
[root@ip-172-31-109-235 docker-tasks]# mv NodeJs_based_source node/
[root@ip-172-31-109-235 docker-tasks]# ls -R
```

in that docker-tasks you need to create sub directories

- mkdir frontend java node

- mv Frontend\_based\_source frontend/
- mv Java\_based\_source java/
- mv NodeJs\_based\_source node/

```
[root@ip-172-31-109-235 docker-tasks]# mkdir frontend
mkdir java
mkdir node
[root@ip-172-31-109-235 docker-tasks]# mv Frontend_based_source frontend/
[root@ip-172-31-109-235 docker-tasks]# mv Java_based_source java/
[root@ip-172-31-109-235 docker-tasks]# mv NodeJs_based_source node/
[root@ip-172-31-109-235 docker-tasks]# ls -R
```

You need to have your directories and files in this structure.

**docker-tasks/**

**frontend/**

**Frontend\_based\_source/**

**Dockerfile.frontend**

**java/**

**Java\_based\_source/**

**Dockerfile.java**

**node/**

**NodeJs\_based\_source/**

**Dockerfile.node**

```
[root@ip-172-31-109-235 docker-tasks]# ls -R
.:
README.md  frontend  java  node

./frontend:
Frontend_based_source

./frontend/Frontend_based_source:
index.html  javascript.js  style.css  todayDeal.js

./java:
java_based_source

./java/Java_based_source:
pom.xml  src

./java/Java_based_source/src:
main

./java/Java_based_source/src/main:
java  resources

./java/Java_based_source/src/main/java:
com

./java/Java_based_source/src/main/java/com:
hussain

./java/Java_based_source/src/main/java/com/hussain:
StartApplication.java

./java/Java_based_source/src/main/resources:
application.properties  static  templates
```

Go to frontend directory and create a file

- vi Dockerfile.frontend

**FROM nginx:latest**

**COPY Frontend\_based\_source /usr/share/nginx/html**

**EXPOSE 80**

go to java directory

- vi Dockerfile.java

**FROM maven:3.9.6-eclipse-temurin-17 AS build**

**WORKDIR /app**

```
COPY Java_based_source /app
RUN mvn clean package -DskipTests
```

```
FROM eclipse-temurin:17-jdk
```

```
WORKDIR /app
```

```
COPY --from=build /app/target/*.jar app.jar
```

```
EXPOSE 8080
```

```
ENTRYPOINT ["java", "-jar", "app.jar"]
```

go to node directory

- vi Dockerfile.node

```
FROM node:18
```

```
WORKDIR /app
```

```
COPY NodeJs_based_source /app
```

```
RUN npm install
```

```
RUN npm run build
```

```
EXPOSE 3000
```

```
CMD ["npm", "start"]
```

```
[root@ip-172-31-109-235 docker-tasks]# cd frontend
[root@ip-172-31-109-235 frontend]# vi Dockerfile.frontend
[root@ip-172-31-109-235 frontend]# cd ../java
[root@ip-172-31-109-235 java]# vi Dockerfile.java
[root@ip-172-31-109-235 java]# cd ../node
[root@ip-172-31-109-235 node]# vi Dockerfile.node
```

In your git repository your Jenkins file should be like this.

<https://github.com/mujaheed00/Python-app.git>

```
pipeline {
```

```
    agent any
```

```
    environment {
```

```
        DOCKERHUB_CREDENTIALS = credentials('dockerhub-creds')
```

```
        IMAGE_NAME = "mujaheed00/python-app"
```

```
}
```

```
stages {
```

```
    stage('Clone Repository') {
```

```
        steps {
```

```
            echo "Cloning repo..."
```

```
            git branch: 'main',
```

```
            url: 'https://github.com/mujaheed00/Python-app.git'
```

```
}
```

```
}
```

```
stage('Build Docker Image') {  
    steps {  
        echo "Building Docker image..."  
        sh """"  
            docker build -t ${IMAGE_NAME}:latest .  
        """"  
    }  
}
```

```
stage('Docker Login') {  
    steps {  
        echo "Logging into Docker Hub..."  
        sh """"  
            echo ${DOCKERHUB_CREDENTIALS_PSW} |  
            docker login -u ${DOCKERHUB_CREDENTIALS_USR} --  
            password-stdin  
        """"  
    }  
}
```

```
stage('Push Image') {  
    steps {  
        echo "Pushing Docker image..."  
        sh """"  
            docker push ${IMAGE_NAME}:latest  
        """"  
    }  
}  
  
post {  
    always {  
        script {  
            echo "Pipeline finished!"  
            sh "docker images"  
        }  
    }  
}  
}
```

Go to your job select scm and give your repository.

Not secure 54.162.209.181:8080/job/Build-and-Push-Docker/configure

Gmail YouTube Maps

Jenkins / Build-and-Push-Docker / Configuration

Configure

General Triggers Pipeline Advanced

SCM ?

Git

Repositories ?

Repository URL ?  
https://github.com/mujaheed00/Python-app.git

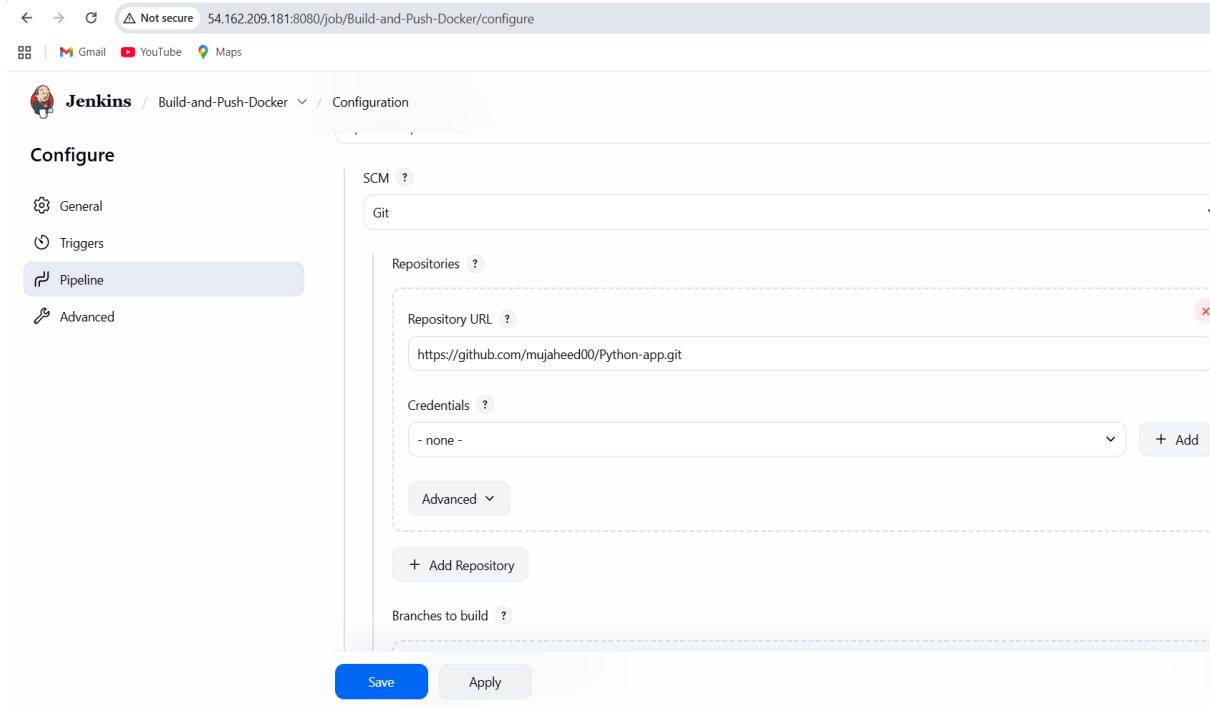
Credentials ? - none - + Add

Advanced

+ Add Repository

Branches to build ?

Save Apply



Click on build now.

Not secure 54.162.209.181:8080/job/Build-and-Push-Docker/

Gmail YouTube Maps

Jenkins / Build-and-Push-Docker

Status Build-and-Push-Docker

</> Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Stages

Rename

Pipeline Syntax

Credentials

Stage View

Average stage times: (full run time: ~26s)

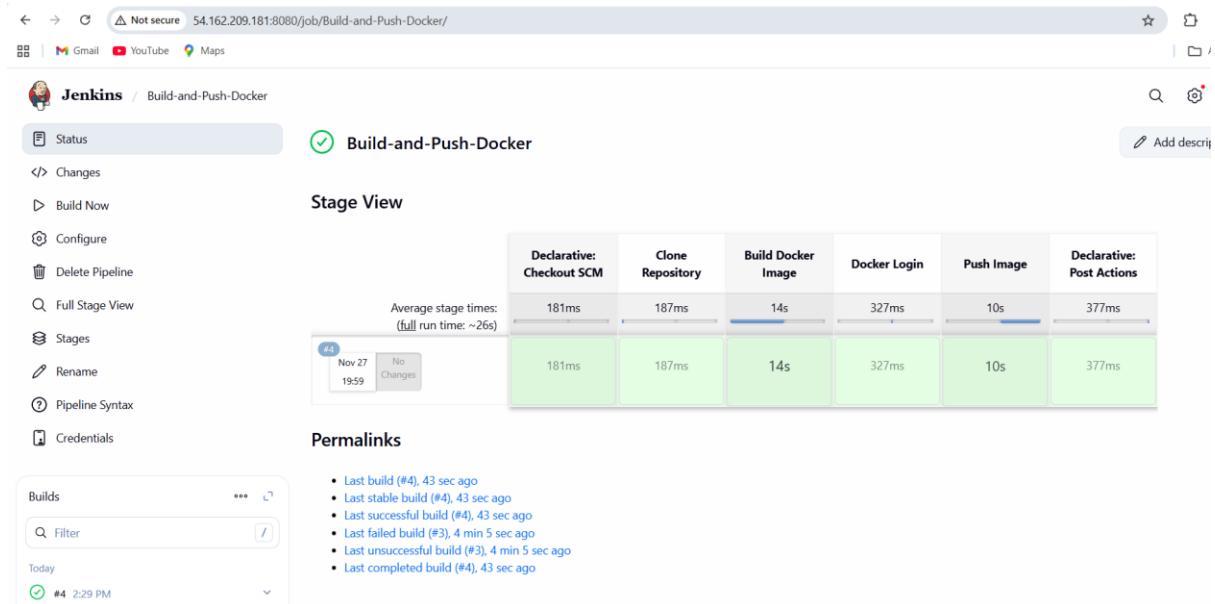
Declarative: Checkout SCM	Clone Repository	Build Docker Image	Docker Login	Push Image	Declarative: Post Actions
181ms	187ms	14s	327ms	10s	377ms
Nov 27 19:59	No Changes	181ms	187ms	14s	327ms
		10s			377ms

Permalinks

- Last build (#4), 43 sec ago
- Last stable build (#4), 43 sec ago
- Last successful build (#4), 43 sec ago
- Last failed build (#3), 4 min 5 sec ago
- Last unsuccessful build (#3), 4 min 5 sec ago
- Last completed build (#4), 43 sec ago

Builds \*\*\*

Filter Today #4 2:29 PM



An repository and image has been created in your dockerhub.

[hub.docker.com/repositories/mujaheed00](https://hub.docker.com/repositories/mujaheed00)

hub Explore My Hub Search Docker Hub Ctrl

**Repositories**  
All repositories within the **mujaheed00** namespace.

Name	Last Pushed	Contains
mujaheed00/python-app	2 minutes ago	IMAGE
mujaheed00/my-custom-image	about 22 hours ago	IMAGE
mujaheed00/myapp	1 day ago	IMAGE
mujaheed00/nginx-sample	1 day ago	IMAGE

1–4 of 4 < >

[hub.docker.com/repository/docker/mujaheed00/python-app/general](https://hub.docker.com/repository/docker/mujaheed00/python-app/general)

hub Explore My Hub Search Docker Hub Ctrl

**mujaheed00/python-app** ⓘ  
Last pushed 2 minutes ago • ⭐0 • ⏺0

Add a description ⓘ ⓘ Add a category ⓘ ⓘ

**General** Tags Image Management BETA Collaborators Webhooks Settings

Tag	OS	Type	Pulled	Pushed
latest	🐧	Image	less than 1 day	2 minutes

[See all](#)

**Repository overview** ⓘ INCOMPLETE

**Build with Docker** ⓘ  
Accelerate and share your Docker Build Infrastructure  
Get faster multi-platform management

```
[root@ip-172-31-109-235 ~]# cd /var/lib/jenkins
[root@ip-172-31-109-235 jenkins]# vi docker-compose.yml
[root@ip-172-31-109-235 jenkins]# docker-compose up -d
```

- vi docker-compose.yml

```

version: '3.8'

services:
  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: root123
      MYSQL_DATABASE: wpdb
      MYSQL_USER: wpuser
      MYSQL_PASSWORD: wppass
    volumes:
      - db_data:/var/lib/mysql

  wordpress:
    image: wordpress:latest
    restart: always
    ports:
      - "8080:80"
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wpuser
      WORDPRESS_DB_PASSWORD: wppass
      WORDPRESS_DB_NAME: wpdb
    depends_on:
      - db

volumes:
  db_data:

~

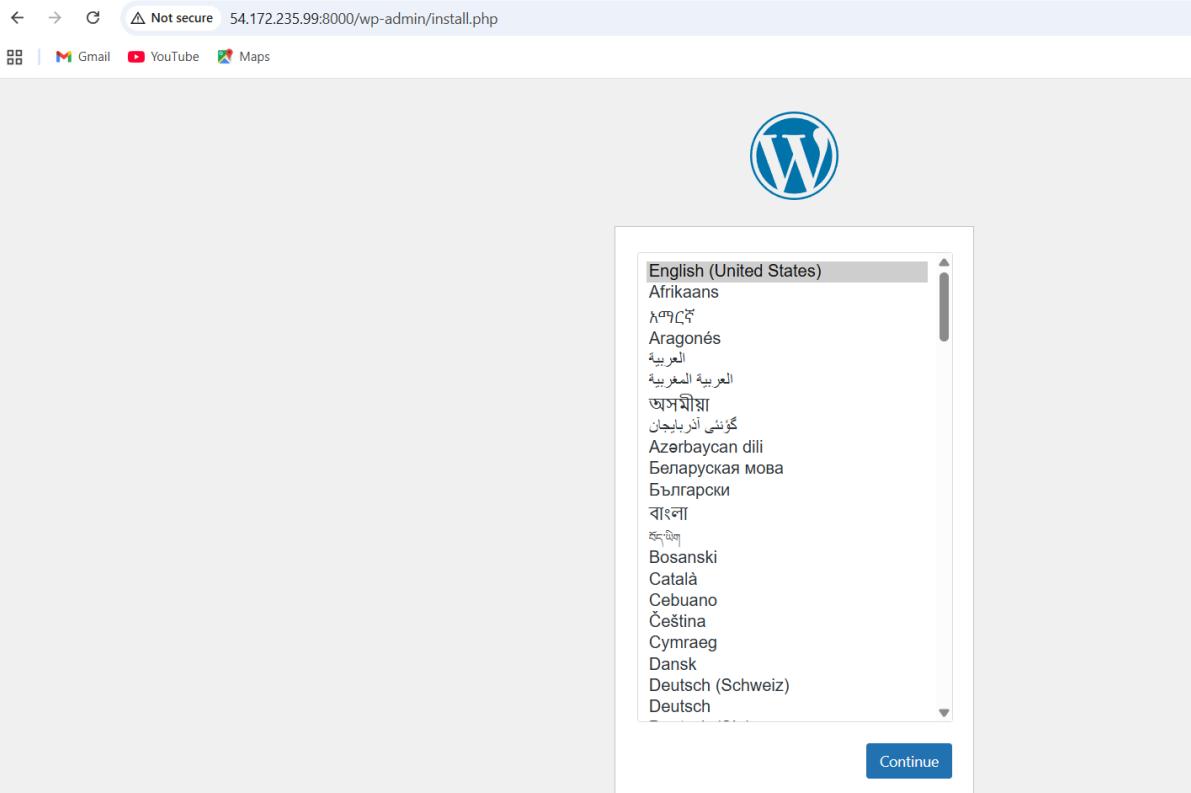
```

- docker-compose up -d

```

[root@ip-172-31-109-235 jenkins]# sudo chmod +x /usr/libexec/docker/cli-plugins/docker-compose
[root@ip-172-31-109-235 jenkins]# docker compose up -d
WARN[0000] /var/lib/jenkins/docker-compose.yml: the attribute `version` is obsolete, it will be ignore
void potential confusion
[+] Running 37/2
  ✓ wordpress Pulled
  ✓ db Pulled
[+] Running 3/4
  ✓ Network jenkins_default          Created
  ✓ Volume "jenkins_db_data"         Created
  ✓ Container jenkins-db-1           Started
  ✘ Container jenkins-wordpress-1   Starting
Error response from daemon: driver failed programming external connectivity on endpoint jenkins_wordp

```



The screenshot shows a web browser window with the URL 54.172.235.99:8000/wp-admin/install.php?step=1. The page title is 'Welcome'. A sub-section titled 'Information needed' asks for site information. It includes fields for 'Site Title' (filled with 'Techie-horizon'), 'Username' (filled with 'mujaheed'), 'Password' (filled with 'Mujaheed2#'), 'Confirm Password' (with a checked checkbox for 'Confirm use of weak password'), 'Your Email' (filled with 'xyz@gmail.com'), and 'Search engine' (with an unchecked checkbox for 'Discourage search engines from indexing this site'). A note below the password field states: 'Important: You will need this password to log in. Please store it in a secure location.' The password field has a red background and a 'Very weak' strength indicator. A 'Hide' link is also present next to the password field.

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

**Site Title** Techie-horizon

**Username** mujaheed

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

**Password** Mujaheed2# Hide Very weak

**Confirm Password**  Confirm use of weak password

**Your Email** xyz@gmail.com

Double-check your email address before continuing.

**Search engine**  Discourage search engines from indexing this site

# Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

## Information needed

Please provide the following information. Do not worry, you can always change these settings later.

**Site Title**

**Username**   
Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

**Password**  Hide Very weak

**Confirm Password**  Confirm use of weak password

**Your Email**   
Double-check your email address before continuing.

**Search engine**  Discourage search engines from indexing this site