

Week 12: Creation of virtual machine for Ubuntu OS and Deploying the web application

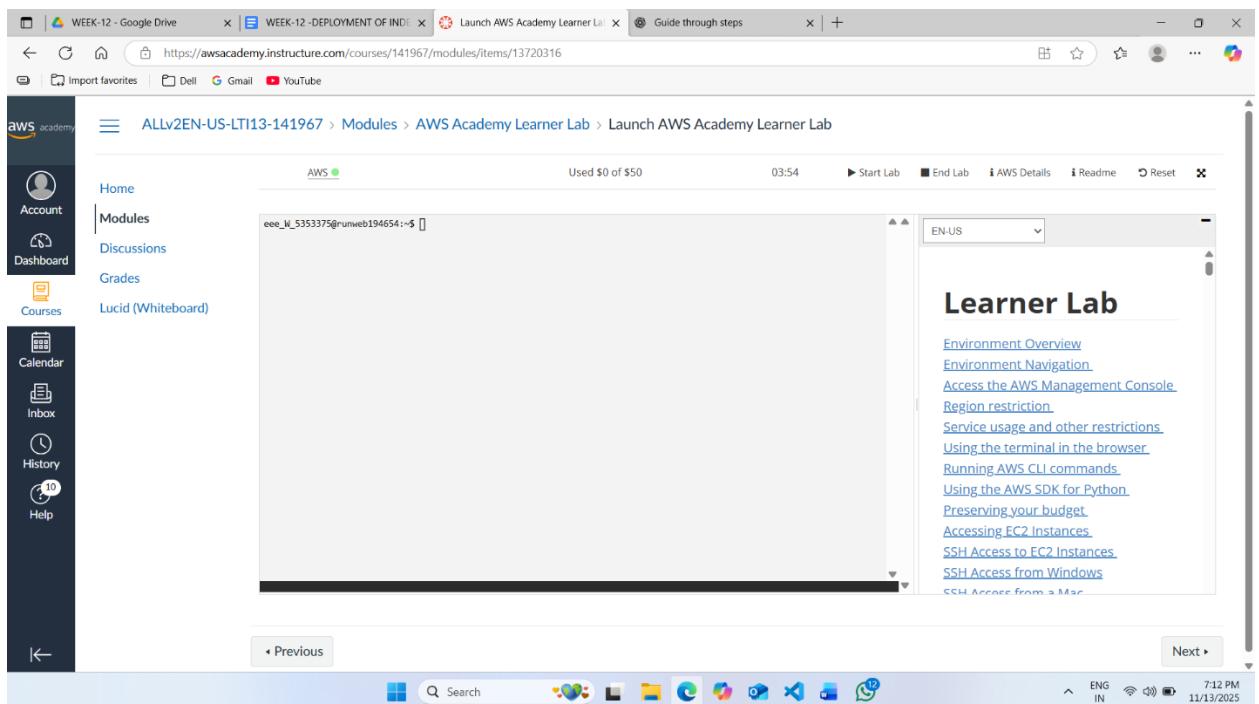
1. Evaluation of previous experiments
 - a. Creating, running and scaling pods in minikube (eg. nginx)
 - b. Running Nagios server and Understanding the Monitoring tool using Docker.
 - c. Creation of virtual machine for Ubuntu OS and deploying an web server
(i.e. DEPLOYMENT of index.html using EC2 instance in AWS)
 - d. Deploying an Maven web project application into cloud
 2. Creation of virtual machine
 3. Deploying the web application
 4. Accessing it publicly
-

Deploying an application into cloud

Steps for Deploying application into the cloud

- I. Create application and Push into github
- II. Create the virtual machine and connect to it.
- III. Clone the application from github, Write the Dockerfile
- IV. Create the image
- V. Run the image and access it public ip of virtual machine

I. Create Maven-web-java project in eclipse & push into github



Click on EC2 to create instance

Stage 1 --Name (Giving name to the machine) ubuntu

Name and tags

ubuntu-mywebserver

Application and OS Images (Amazon Machine Image)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type

Summary

Number of instances: 1

Software Image (AMI): Ubuntu Server 22.04 LTS (HVM), ami-0c398cb65a93047f2

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

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

Launch instance

Stage 2 -- Select AMI (Note: Select free tier eligible) ubuntu server

Stage 3 -- Architecture as 64-bit

AMI from catalog

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type

Description

Ubuntu Server 22.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Summary

Number of instances: 1

Software Image (AMI): Ubuntu Server 22.04 LTS (HVM), ami-0c398cb65a93047f2

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

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

Launch instance

Stage 4 -- Instance type ---- t2.micro(default 1 CPU,1 GB RAM)

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t2.micro
 Family: t2 1 vCPU 1 GiB Memory Current generation: true
 On-Demand Windows base pricing: 0.0162 USD per Hour
 On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
 On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.026 USD per Hour
 On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

[Compare instance types](#)

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

Stage 5 -- Create a new keypair---a keypair will download with extension .pem

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA RSA encrypted private and public key pair

ED25519 ED25519 encrypted private and public key pair

Private key file format

.pem For use with OpenSSH

.ppk For use with PuTTY

▼ 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

[Create new key pair](#)

▼ Network settings [Info](#)

Network | [Info](#)

[Edit](#)

Stage 6 -- Network Setting ----Create Security group -- (It deals with ports)

Network settings

- Network: vpc-01281b3c50636b0e4
- Subnet: No preference (Default subnet in any availability zone)
- Auto-assign public IP: Enabled
- Firewall (security groups):
 - Create security group
 - Select existing security group
- We'll create a new security group called 'launch-wizard-1' with the following rules:
 - Allow SSH traffic from Anywhere (0.0.0.0/0)
 - Allow HTTPS traffic from the internet (To set up an endpoint, for example when creating a web server)
 - Allow HTTP traffic from the internet (To set up an endpoint, for example when creating a web server)

Summary

- Number of instances: 1
- Software Image (AMI): Ubuntu Server 22.04 LTS (HVM) ...read more
- Virtual server type (instance type): t2.micro
- Key pair (login): my-lab-key
- Storage (volume): 1 volume(s) - 8 GiB

Stage 7 -- Storage - 8GB (Observation - we have root - it is same as C Drive) it default 8 GB

Configure storage

- 1x 8 GiB gp2 Root volume, Not encrypted
- Add new volume

The selected AMI contains instance store volumes, however the instance does not allow any instance store volumes. None of the instance store volumes from the AMI will be accessible from the instance.

Click refresh to view backup information

The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

0 x File systems

Advanced details

Summary

- Number of instances: 1
- Software Image (AMI): Ubuntu Server 22.04 LTS (HVM) ...read more
- Virtual server type (instance type): t2.micro
- Key pair (login): my-lab-key
- Storage (volume): 1 volume(s) - 8 GiB

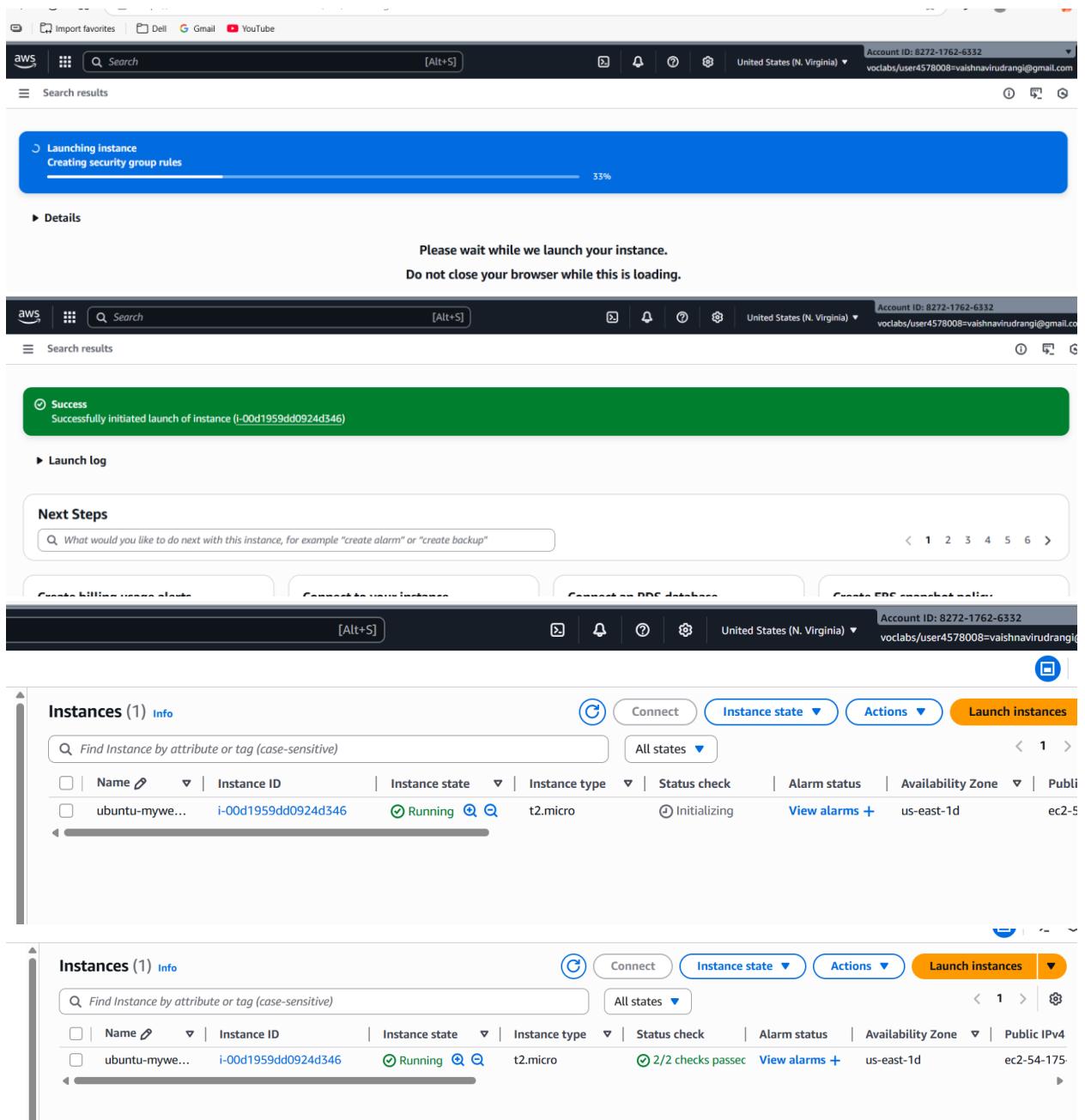
Stage 8 --- click on launch instance

The screenshot shows a CloudFormation console interface. At the top, there's a header bar with the account ID "8272-1762-6332" and email "voclabs/user4578008=vaishnavirudrangi@gmail.com". Below the header, a navigation bar includes icons for "Instances", "CloudWatch Metrics", "Logs", and "CloudWatch Metrics Insights".

The main content area is titled "▼ Summary". It displays the following configuration details:

- Number of instances**: 1
- Software Image (AMI)**: Ubuntu Server 22.04 LTS (HVM),... [read more](#)
ami-0c398cb65a93047f2
- Virtual server type (instance type)**: t2.micro
- Firewall (security group)**: New security group
- Storage (volumes)**: 1 volume(s) - 8 GiB

At the bottom of the summary card, there are three buttons: "Cancel", "Launch instance" (which is highlighted in orange), and "Preview code".



Step a: Now you can connect local system to server (EC2 instance) using secure shell SSH.

aws | Search [Alt+S] | United States (N. Virginia) | Account ID: 8272-1762-6532
EC2 > Instances > i-00d1959dd0924d346 > Connect to instance

Connect Info

Connect to an instance using the browser-based client.

EC2 Instance Connect | Session Manager | **SSH client** | EC2 serial console

Instance ID
i-00d1959dd0924d346 (ubuntu-mywebserver)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is my-lab-key.pem.
3. Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "my-lab-key.pem"
4. Connect to your instance using its Public DNS:
ssh -i "my-lab-key.pem" ubuntu@ec2-54-175-29-82.compute-1.amazonaws.com

Command copied

ssh -i "my-lab-key.pem" ubuntu@ec2-54-175-29-82.compute-1.amazonaws.com

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

Step b: Open Powershell in administrative mode and navigate to that path.

Type: cd < path>

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> cd C:\Users\vaishnavi\OneDrive\Desktop\AWS
PS C:\Users\vaishnavi\OneDrive\Desktop\AWS>
```

Step c: Go to SSh and copy the command ssh which is present at the below

Step d: Run the pasted ssh -i command in the terminal

```
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> cd C:\Users\vaishnavi\OneDrive\Desktop\AWS
PS C:\Users\vaishnavi\OneDrive\Desktop\AWS> ssh -i "my-lab-key.pem" ubuntu@ec2-54-175-29-82.compute-1.amazonaws.com
The authenticity of host 'ec2-54-175-29-82.compute-1.amazonaws.com (54.175.29.82)' can't be established.
ED25519 key fingerprint is SHA256:7z+M9BUhiQrmP8boQPGpza7EPBCHDqnP+ogA8CPuxQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? 
```

```
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-172-31-24-34:~$
```

Run the following commands to install s/w

```
ubuntu@ip-172-31-24-34:~$ sudo apt update  
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]  
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]  
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]  
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]  
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]  
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]  
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]  
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]  
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [3076 kB]  
Get:12 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [2816 kB]  
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [475 kB]  
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [19.0 kB]  
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [4853 kB]  
Get:16 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [408 kB]  
Get:17 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [13.9 kB]  
Get:18 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [4705 kB]  
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [903 kB]  
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [640 B]  
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1242 kB]  
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [308 kB]  
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [29.8 kB]  
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [57.6 kB]  
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [13.2 kB]  
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [600 B]  
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [69.4 kB]  
Get:28 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [11.5 kB]
```

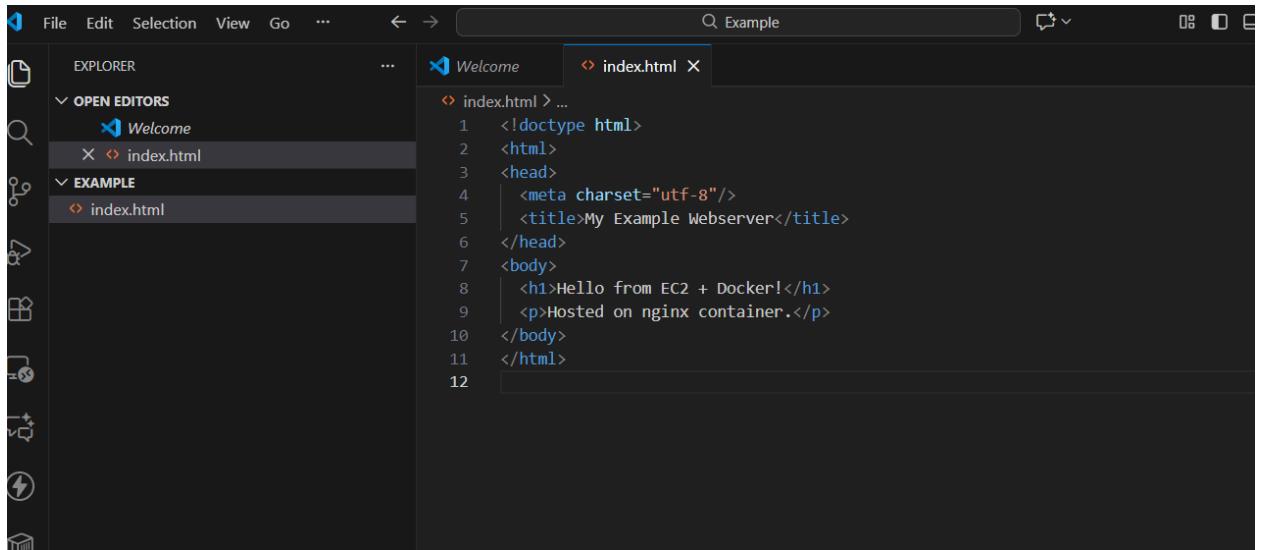
```
19 packages can be upgraded. Run 'apt list --upgradable' to see them.  
ubuntu@ip-172-31-24-34:~$ sudo apt-get install docker.io  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan  
Suggested packages:  
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx docker-compose-v2 docker-doc rinse  
  zfs-fuse | zfsutils  
The following NEW packages will be installed:  
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan  
0 upgraded, 8 newly installed, 0 to remove and 19 not upgraded.  
Need to get 76.3 MB of archives.  
After this operation, 289 MB of additional disk space will be used.  
Do you want to continue? [Y/n] Y  
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1 [63.6 kB]  
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 bridge-utils amd64 1.7-1ubuntu3 [34.4 kB]  
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 runc amd64 1.3.3-0ubuntu1~22.04.2 [8856 kB]  
]  
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 containerd amd64 1.7.28-0ubuntu1~22.04.1 [38.5 MB]  
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 dns-root-data all 2024071801~ubuntu0.22.04.1 [6132 B]  
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 dnsmasq-base amd64 2.90-0ubuntu0.22.04.1 [374 kB]  
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 docker.io amd64 28.2.2-0ubuntu1~22.04.1 [28.4 MB]  
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]  
Fetched 76.3 MB in 1s (92.6 MB/s)
```

```
ubuntu@ip-172-31-24-34:~$ sudo apt install git  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
git is already the newest version (1:2.34.1-1ubuntu1.15).  
git set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 19 not upgraded.  
ubuntu@ip-172-31-24-34:~$
```

```
ubuntu@ip-172-31-24-34:~$ sudo apt install nano  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
nano is already the newest version (6.2-1ubuntu0.1).  
nano set to manually installed.  
0 upgraded, 0 newly installed, 0 to remove and 19 not upgraded.  
ubuntu@ip-172-31-24-34:~$
```

Now we want to create an application, push it into git, create docker image of it and run it

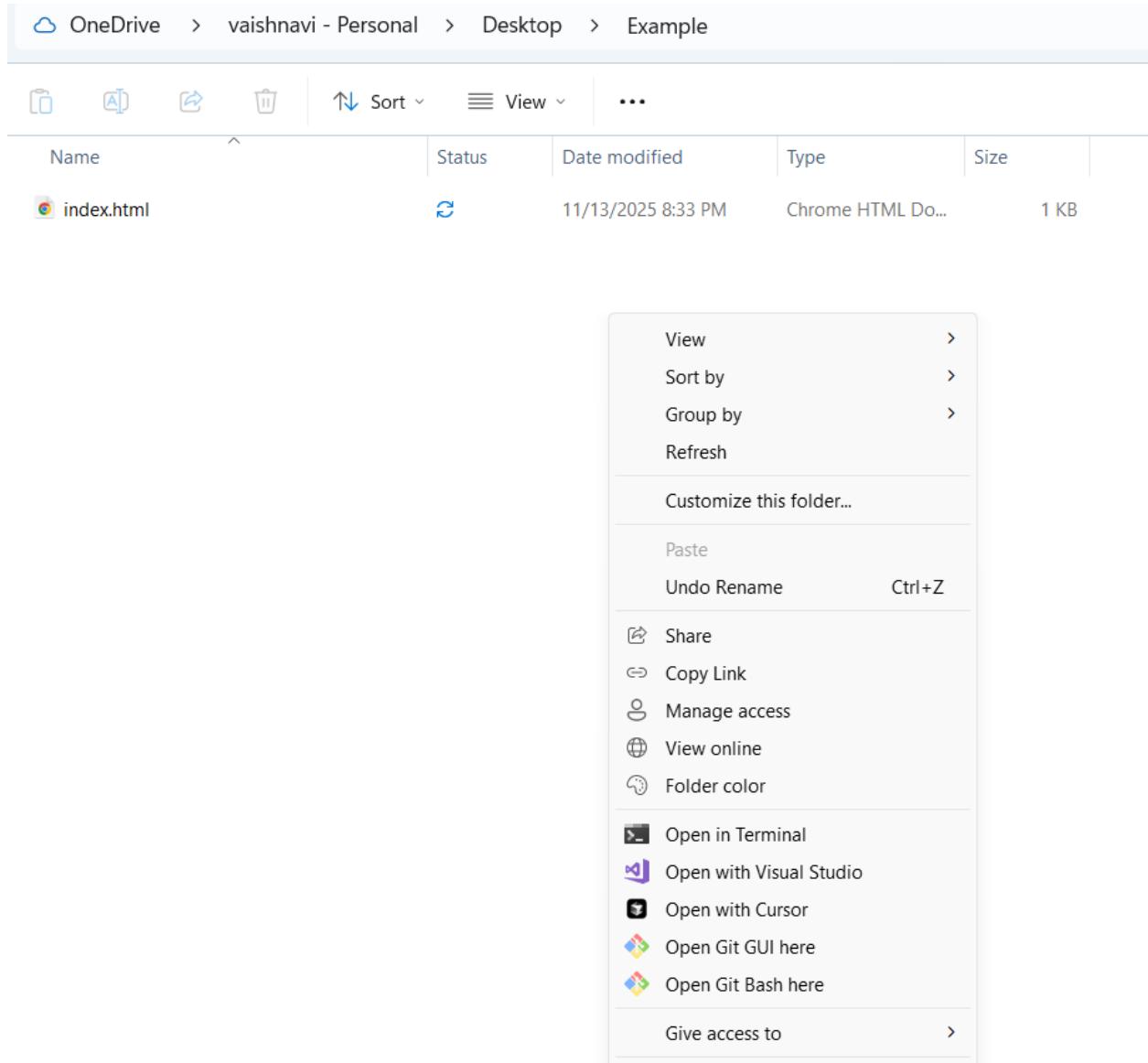
Step a: Create basic index.html file in folder Example and save it



The screenshot shows the Visual Studio Code interface. The left sidebar has sections for 'EXPLORER', 'OPEN EDITORS', and 'EXAMPLE'. In the 'EXAMPLE' section, there is a file named 'index.html'. The right side shows an open editor for 'index.html' with the following content:

```
1 <!doctype html>
2 <html>
3 <head>
4   <meta charset="utf-8"/>
5   <title>My Example Webserver</title>
6 </head>
7 <body>
8   <h1>Hello from EC2 + Docker!</h1>
9   <p>Hosted on nginx container.</p>
10 </body>
11 </html>
```

Step b: Open git Bash in folder Example by right clicking with mouse

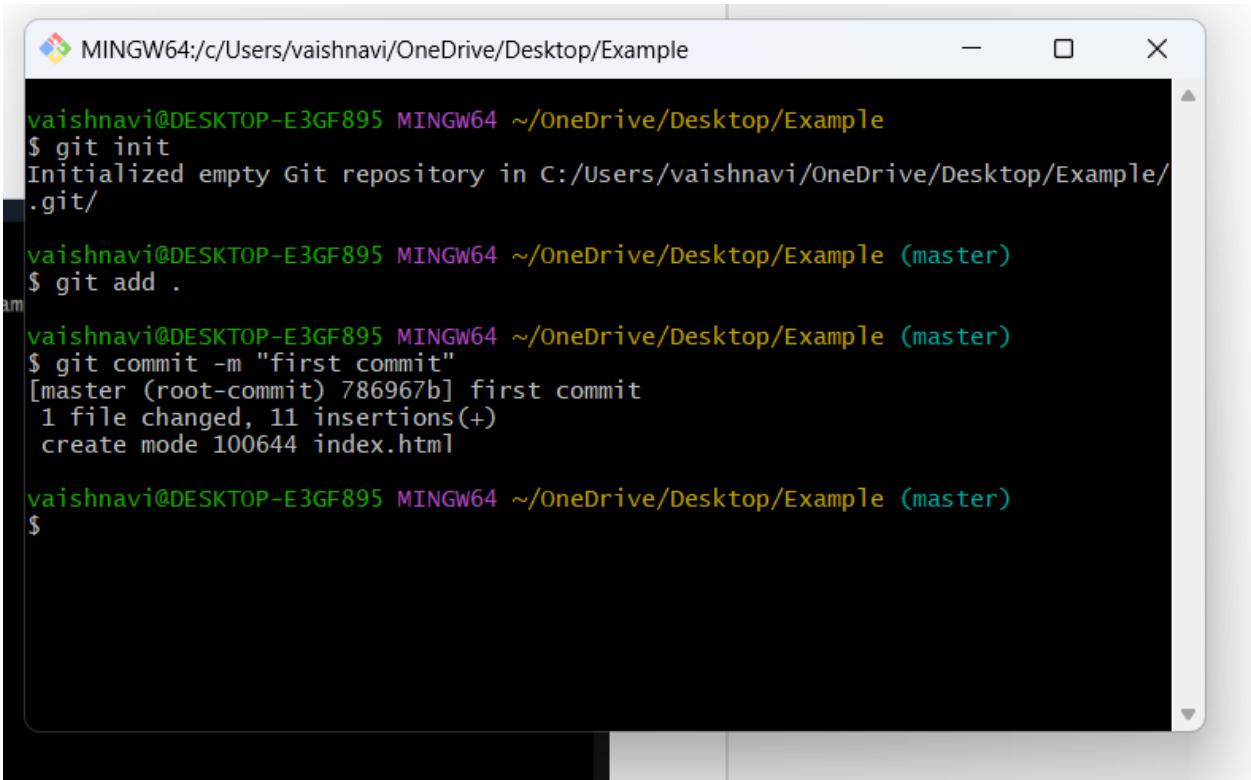


Step c: In git bash run the following commands

git init

git add .

Git commit -m "first commit"



```
MINGW64:/c/Users/vaishnavi/OneDrive/Desktop/Example
vaishnavi@DESKTOP-E3GF895 MINGW64 ~/OneDrive/Desktop/Example
$ git init
Initialized empty Git repository in C:/Users/vaishnavi/OneDrive/Desktop/Example/.git/
vaishnavi@DESKTOP-E3GF895 MINGW64 ~/OneDrive/Desktop/Example (master)
$ git add .
vaishnavi@DESKTOP-E3GF895 MINGW64 ~/OneDrive/Desktop/Example (master)
$ git commit -m "first commit"
[master (root-commit) 786967b] first commit
 1 file changed, 11 insertions(+)
 create mode 100644 index.html
vaishnavi@DESKTOP-E3GF895 MINGW64 ~/OneDrive/Desktop/Example (master)
$
```

Step d: Create git repository (here with name AWS)

Step e: Copy command one by one from above repository and run as below

git branch -M main

git remote add origin <https url>

git push -u origin main

```
vaishnavi@DESKTOP-E3GF895 MINGW64 ~/OneDrive/Desktop/Example (master)
$ git branch -M main
```

```
vaishnavi@DESKTOP-E3GF895 MINGW64 ~/OneDrive/Desktop/Example (main)
$ git remote add origin https://github.com/Rudrangi-Vaishnavi/AWS.git
```

```
vaishnavi@DESKTOP-E3GF895 MINGW64 ~/OneDrive/Desktop/Example (main)
$ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 439 bytes | 219.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Rudrangi-Vaishnavi/AWS.git
  c4e5a44..8d85707 main -> main
branch 'main' set up to track 'origin/main'.
```

Step f: refresh repository and You can now see index.html in github

The screenshot shows a GitHub repository page for a repository named 'AWS'. The repository is public and has two commits. The first commit was made by 'Rudrangi-Vaishnavi' and is labeled 'first commit'. The second commit is labeled 'Initial commit'. Both commits were made 1 minute ago. The 'index.html' file contains the text 'AWS'. On the right side of the page, there is an 'About' section with a note that there is no description, and a sidebar with options like 'Readme', 'Activity', 'Stars', 'Watching', and 'Forks'. There are also sections for 'Releases' and 'Create a new release'.

Step g: Copy http path

Step h: Clone the repository with copied http path by command in command prompt

```
git clone <copied http url>
```

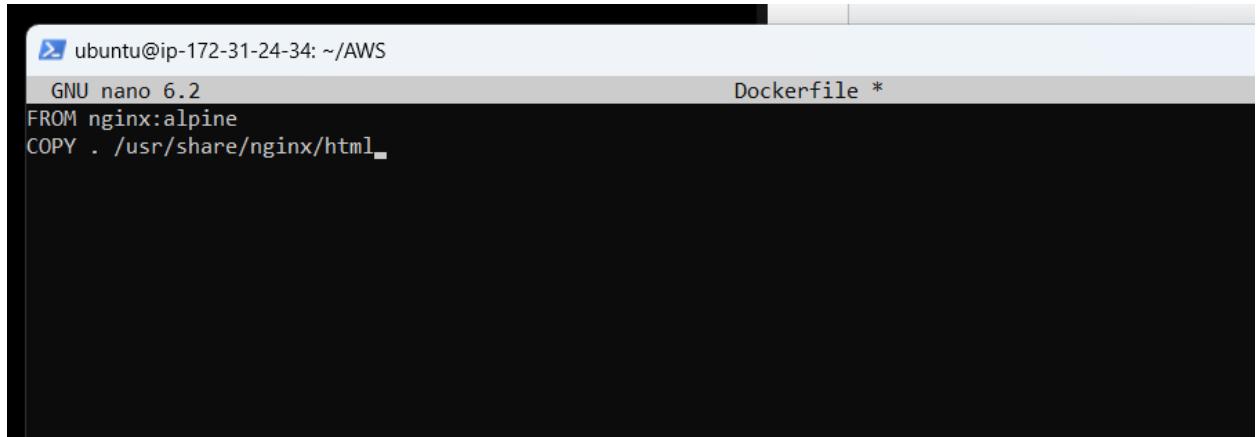
```
0 upgraded, 0 newly installed, 0 to remove and 19 not upgraded.  
ubuntu@ip-172-31-24-34:~$ git clone https://github.com/Rudrangi-Vaishnavi/AWS.git  
Cloning into 'AWS'...  
remote: Enumerating objects: 6, done.  
remote: Counting objects: 100% (6/6), done.  
remote: Compressing objects: 100% (4/4), done.  
remote: Total 6 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)  
Receiving objects: 100% (6/6), done.  
ubuntu@ip-172-31-24-34:~$
```

Step i: Navigate to the cloned folder. Type cd AWS as below, next ls to

```
remote: Total 6 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)  
Receiving objects: 100% (6/6), done.  
ubuntu@ip-172-31-24-34:~$ ls  
AWS  
ubuntu@ip-172-31-24-34:~$ cd AWS  
ubuntu@ip-172-31-24-34:~/AWS$ ls  
README.md  index.html  
ubuntu@ip-172-31-24-34:~/AWS$
```

Step j: create Dockerfile in above command prompt in ububtu ie in power shell

Step k: Write the following data in Dockerfile and click ctrl+o Enter and then ctrl-x

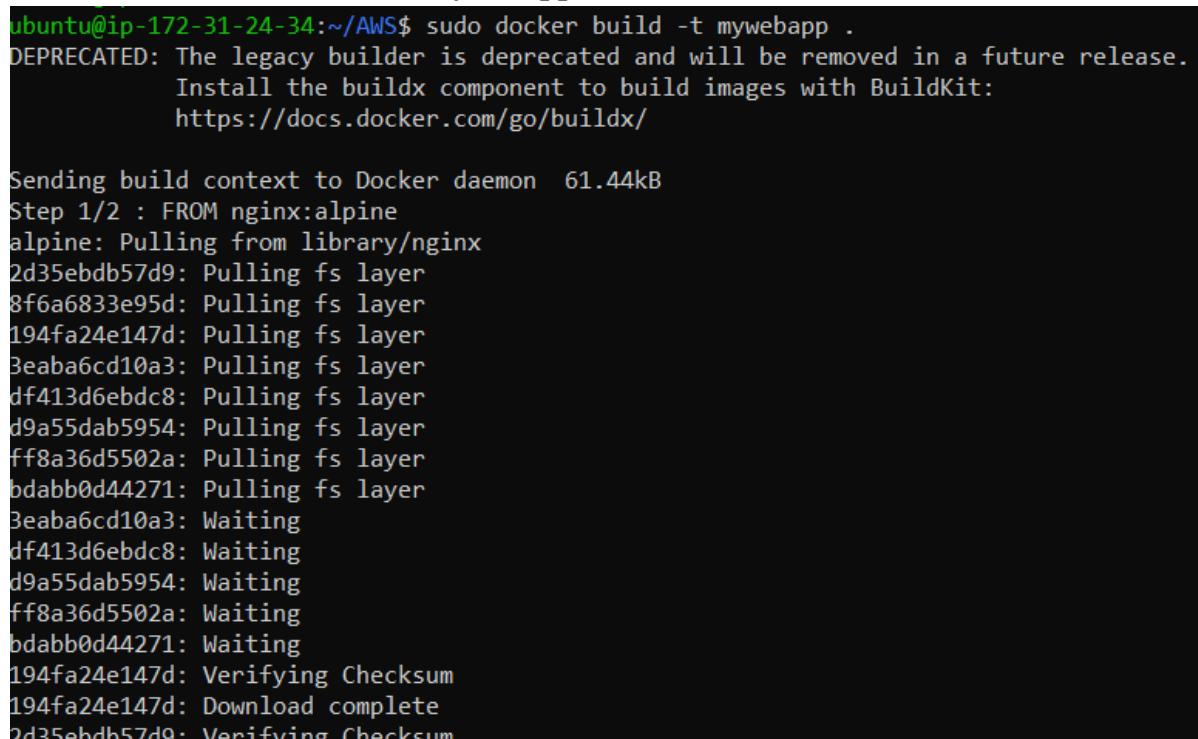


A screenshot of a terminal window titled "ubuntu@ip-172-31-24-34: ~/AWS". The window shows the command "nano Dockerfile" being run, followed by the Dockerfile content:

```
GNU nano 6.2
FROM nginx:alpine
COPY . /usr/share/nginx/html
```

Step L: Build docker image by executing the following command

sudo docker build -t mywebapp



A screenshot of a terminal window titled "ubuntu@ip-172-31-24-34:~/AWS\$". The command "sudo docker build -t mywebapp ." is run, followed by the output of the build process:

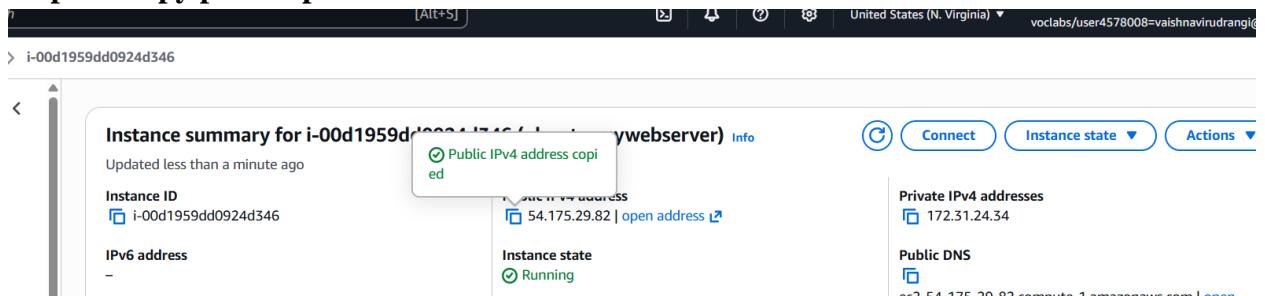
```
sudo docker build -t mywebapp .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 61.44kB
Step 1/2 : FROM nginx:alpine
alpine: Pulling from library/nginx
2d35ebdb57d9: Pulling fs layer
8f6a6833e95d: Pulling fs layer
194fa24e147d: Pulling fs layer
3eaba6cd10a3: Pulling fs layer
df413d6ebdc8: Pulling fs layer
d9a55dab5954: Pulling fs layer
ff8a36d5502a: Pulling fs layer
bdabb0d44271: Pulling fs layer
3eaba6cd10a3: Waiting
df413d6ebdc8: Waiting
d9a55dab5954: Waiting
ff8a36d5502a: Waiting
bdabb0d44271: Waiting
194fa24e147d: Verifying Checksum
194fa24e147d: Download complete
2d35ebdb57d9: Verifying Checksum
```

Step m: run the image and map it to port 80

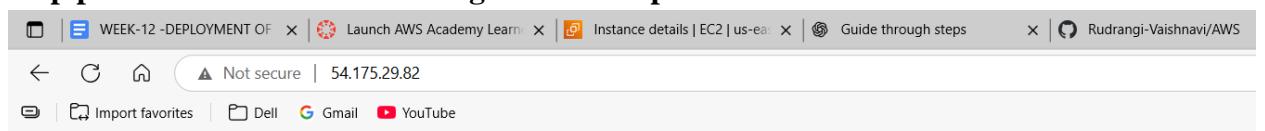
```
Status: Downloaded newer image for nginx:alpine
---> d4918ca78576
Step 2/2 : COPY . /usr/share/nginx/html
---> 41a8b396d4e1
Successfully built 41a8b396d4e1
Successfully tagged mywebapp:latest
ubuntu@ip-172-31-24-34:~/AWS$ sudo docker run -d -p 80:80 mywebapp
6d01321d73d4d21c80ba9b4697ec01fe63c863590444b203774c306a5a74eb3c
ubuntu@ip-172-31-24-34:~/AWS$
```

Step n: Go to instances and click on instance here

Step o: Copy public ipv4 address



Step p: Paste it in the browser to get below output



Hello from EC2 + Docker!

Hosted on nginx container.

Step Q: stop container

```
--> 41a8b396d4e1
Successfully built 41a8b396d4e1
Successfully tagged mywebapp:latest
ubuntu@ip-172-31-24-34:~/AWS$ sudo docker run -d -p 80:80 mywebapp
6d01321d73d4d21c80ba9b4697ec01fe63c863590444b203774c306a5a74eb3c
ubuntu@ip-172-31-24-34:~/AWS$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
NAMES
6d01321d73d4 mywebapp "/docker-entrypoint...." 2 minutes ago Up 2 minutes 0.0.0.0:80->80/tcp, [::]:80->80/tcp
peaceful_blackwell
ubuntu@ip-172-31-24-34:~/AWS$ sudo docker stop 6d01321d73d4
6d01321d73d4
ubuntu@ip-172-31-24-34:~/AWS$
```

Account ID: 8272-1762-6332
voclabs/user4578008=vaishnavirudrangi@gmail.com

United States (N. Virginia) ▾

Connect Instance state Actions ▾ Launch instances ▾

All states ▾

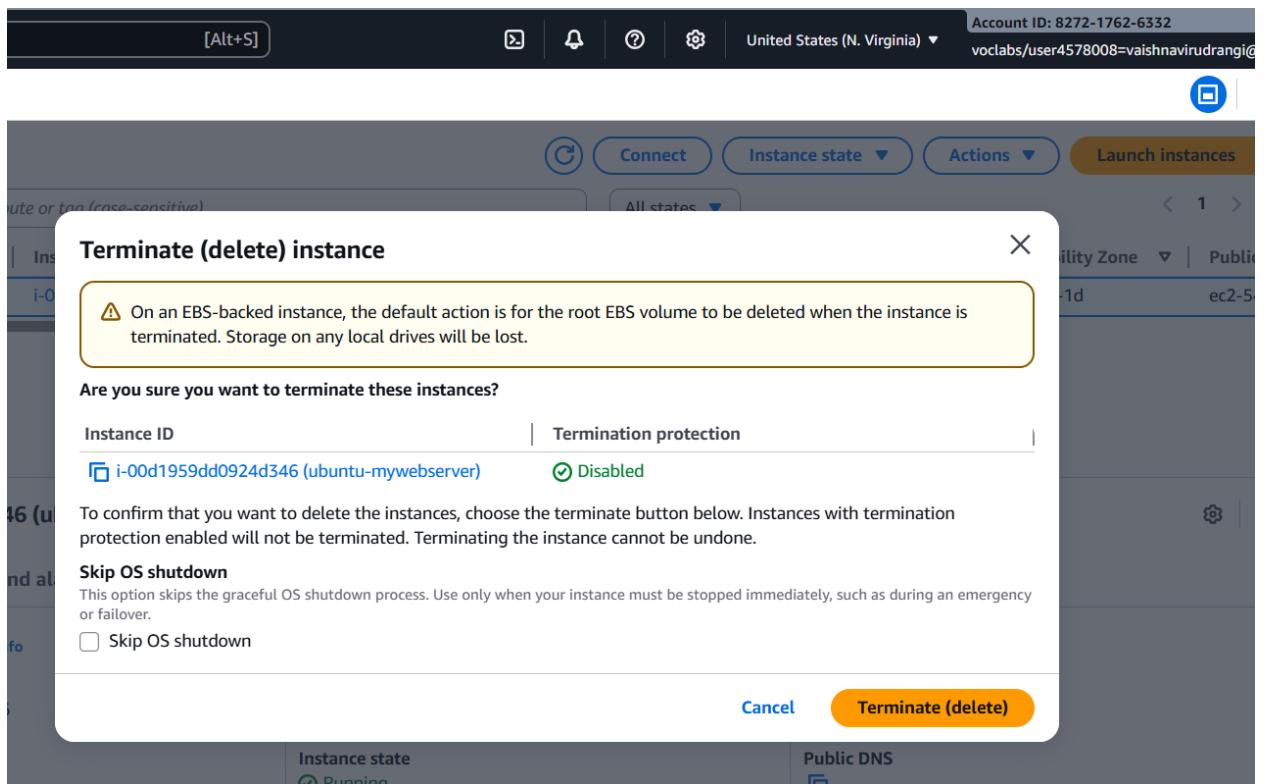
Stop instance Start instance Reboot instance Hibernate instance

Terminate (delete) instance

Availability Zone ▾ Public IPv4

us-east-1d ec2-54-175-

This screenshot shows two separate AWS interfaces. The top part is a terminal window displaying Docker commands and logs. The bottom part is the AWS Lambda console, specifically the 'Actions' section for an instance. The 'Actions' dropdown menu is open, showing options like Stop instance, Start instance, Reboot instance, Hibernate instance, and Terminate (delete) instance. The Lambda function name 'peaceful_blackwell' is visible in the background.



ALLv2EN-US-LTI13-141967 > Modules > AWS Academy Learner Lab > Launch AWS Academy Learner Lab



II. Create the virtual machine (EC2--instance) in aws and connect to

Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster.

Ex : **Launch ubuntu instance**

Step 1: Login to AWS /canvas account

9:58



5G 76



...

Course Invitation

Inbox



AWS Academy 24 Oct

to me ▾



...

You've been invited to participate in a class at AWS Academy . The class is called AWS Academy Learner Lab [141967]. Course role: Student

Name: vaishnavirudrangi@gmail.com

Email: vaishnavirudrangi@gmail.com

Username: none

You'll need to register with Canvas before you can participate in the class.

Get Started



CANVAS

Reply

Forward



The screenshot shows the AWS Academy Learner Lab interface. On the left is a dark sidebar with navigation links: Account, Dashboard, Courses, Calendar, Inbox, History, and Help. The main area has tabs for Home, Modules, Discussions, Grades, and Lucid (Whiteboard). The Home tab is selected, showing a balance of \$0 of \$50, a timer at 03:59, and buttons for Start Lab, End Lab, AWS Details, Readme, and Reset. A large central window displays a terminal session with a blue V-shaped cursor and a red arrow pointing right. To the right of the terminal is a sidebar titled "Learner Lab" containing a list of links: Environment Overview, Environment Navigation, Access the AWS Management Console, Region restriction, Service usage and other restrictions, Using the terminal in the browser, Running AWS CLI commands, Using the AWS SDK for Python, Preserving your budget, Accessing EC2 Instances, SSH Access to EC2 Instances, SSH Access from Windows, and SSH Access from a Mac. At the bottom are "Previous" and "Next" buttons.

This screenshot is identical to the one above, showing the AWS Academy Learner Lab interface. The left sidebar, central terminal window, and right sidebar with links are all present. The terminal window shows a command prompt: eee_l_5353375@runweb194654:~\$. The status bar at the bottom indicates the time as 7:04 PM and the date as 11/13/2025.

The screenshot shows the AWS EC2 Dashboard for the United States (N. Virginia) Region. The left sidebar includes links for Dashboard, Instances (with sub-options like Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager), Images (AMIs, AMI Catalog), and Elastic Block Store (Volumes). The main content area has several sections:

- Resources:** Displays metrics for Instances (running: 0), Auto Scaling Groups (0), Capacity Reservations (0), Dedicated Hosts (0), Elastic IPs (0), Instances (0), Key pairs (1), Load balancers (0), Placement groups (0), Security groups (1), Snapshots (0), and Volumes (0).
- Launch instance:** A section with a "Launch instance" button and a "Migrate a server" link.
- Service health:** Shows the region as United States (N. Virginia) and the status as "This service is operating normally".
- Account attributes:** Includes sections for Default VPC (vpc-01281b3c50636b0e4), Settings (Data protection and security, Allowed AMIs, Zones, EC2 Serial Console, Default credit specification, EC2 console preferences), and Explore AWS (Enable Best Price-Performance with AWS Graviton2, Introducing Spot Blueprints, a Real-Time Template Generator).

Step 2: Services -- EC2

Step 3: Choose region which is near ?

Services -- EC2 --- Launch Instance

The screenshot shows the AWS EC2 Home page. On the left, a sidebar menu includes Dashboard, Instances (with sub-options like Instances, Instance Types, Launch Templates, etc.), Images, and Elastic Block Store. The main content area has several sections: 'Resources' (listing Instances (running) 0, Auto Scaling Groups 0, Capacity Reservations 0, Dedicated Hosts 0, Elastic IPs 0, Instances 0, Key pairs 1, Load balancers 0, Placement groups 0, Security groups 1, Snapshots 0, and Volumes 0), 'Launch instance' (with 'Launch instance' and 'Migrate a server' buttons), 'Service health' (Region: United States (N. Virginia), Status: This service is operating normally), and 'Account attributes' (Default VPC: vpc-01281b3c50636b0e4, Settings, Explore AWS, and Introducing Spot Blueprints). The bottom right corner shows copyright information and a date (© 2025, Amazon Web Services, Inc. or its affiliates. 11/13/2025).

Stage 1 --Name (Giving name to the machine) ubuntu

The screenshot shows the 'Launch an instance' wizard. Step 1: Name and tags. It asks for a Name (MavenWebProjectExample) and provides an 'Add additional tags' button. Step 2: Application and OS Images (Amazon Machine Image) is partially visible below. To the right, a 'Summary' section shows 1 instance, the Software Image (AMI) as Amazon Linux 2023 AMI 2023.9.2..., and the Virtual server type (instance type) as t3.micro.

Stage 2 -- Select AMI (Note: Select free tier eligible) ubuntu server

Stage 3 -- Architecture as 64-bit

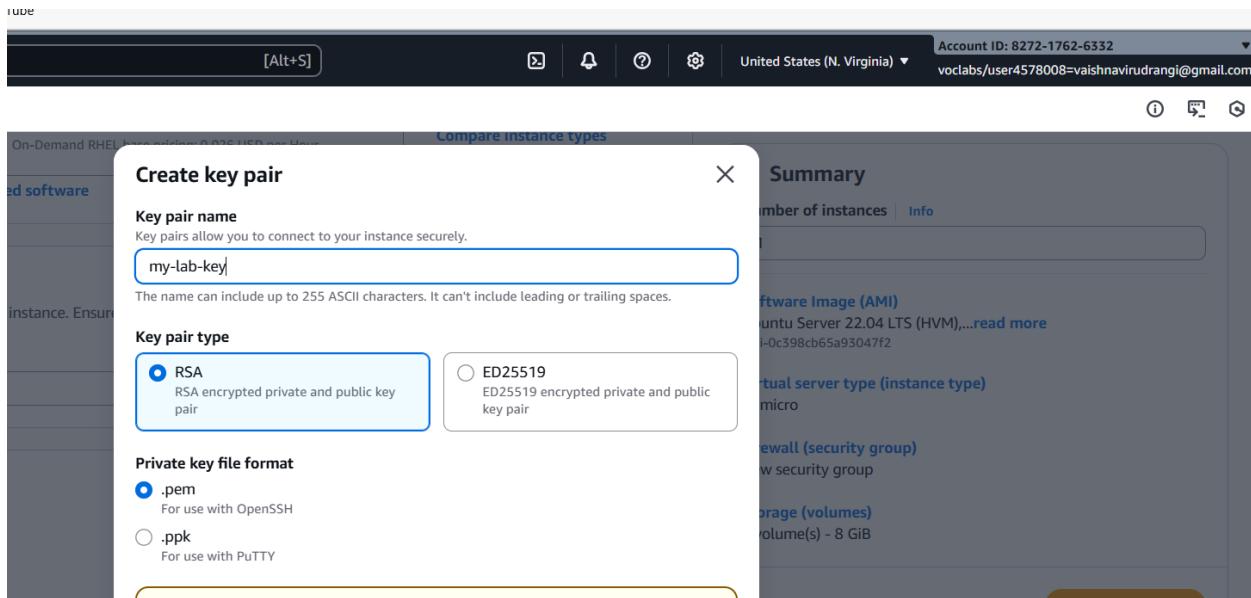
The screenshot shows the AWS CloudFormation console with a stack named 'WEEK-12' in progress. The 'Template' tab is active, displaying the CloudFormation template. The 'Outputs' tab shows an output named 'URL' with the value 'https://d139qzgk3wv4t.cloudfront.net/'. The 'Parameters' tab is also visible.

Stage 4 -- Instance type ---- t2.micro(default 1 CPU,1 GB RAM)

The screenshot shows the AWS Lambda console with a function named 'WEEK-12' in progress. The 'Code' tab is active, showing the Lambda function code. The 'Environment' tab shows environment variables like 'AWS_LAMBDA_FUNCTION_NAME' and 'AWS_LAMBDA_FUNCTION_MEMORY_SIZE'. The 'Logs' tab is also visible.

Stage 5 -- Create a new keypair---a keypair will downloaded with extension .pem

Store key in folder AWS



▼ 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

my-lab-key [▼](#) [Create new key pair](#)

▼ Network settings [Info](#)

[Network](#) [Edit](#)

Stage 6 -- Network Setting ----Create Security group -- (It deals with ports)

(Note for understanding We have 0 to 65535 ports. Every port is dedicated to special purpose)

Do this step : HERE select http and https

The screenshot shows the AWS CloudFormation Launch Wizard interface. On the left, the 'Network settings' step is active, displaying configuration for a VPC (vpc-01281b3c505636b0e4) and a subnet (No preference). It includes options for auto-assigning public IP and creating a security group. A note about firewall rules is present. On the right, the 'Summary' step is shown, detailing the instance type (t2.micro), AMI (Ubuntu Server 22.04 LTS (HVM)), and key pair (my-tut-key). The summary also includes links for network, storage, and advanced details.

Stage 7 -- Storage - 8GB (Observation - we have root - it is same as C Drive)

The screenshot shows the 'Configure storage' step of the launch wizard. It specifies 1x volume of 8 GiB, gp2 type, and 'Root volume, Not encrypted'. An info message notes that the selected AMI contains instance store volumes but the instance does not allow them. Below this, there's a section for backup information and a note about file systems. At the bottom, an 'Advanced details' section is partially visible.

Stage 8 --- click on launch instance

 United States (N. Virginia) ▾ Account ID: 8272-1762-6332
voclabs/user4578008=vaishnavirudrangi@gmail.com

▼ Summary

Number of instances | [Info](#)

1

Software Image (AMI)
Ubuntu Server 22.04 LTS (HVM),...[read more](#)
ami-0c398cb65a93047f2

Virtual server type (instance type)
t2.micro

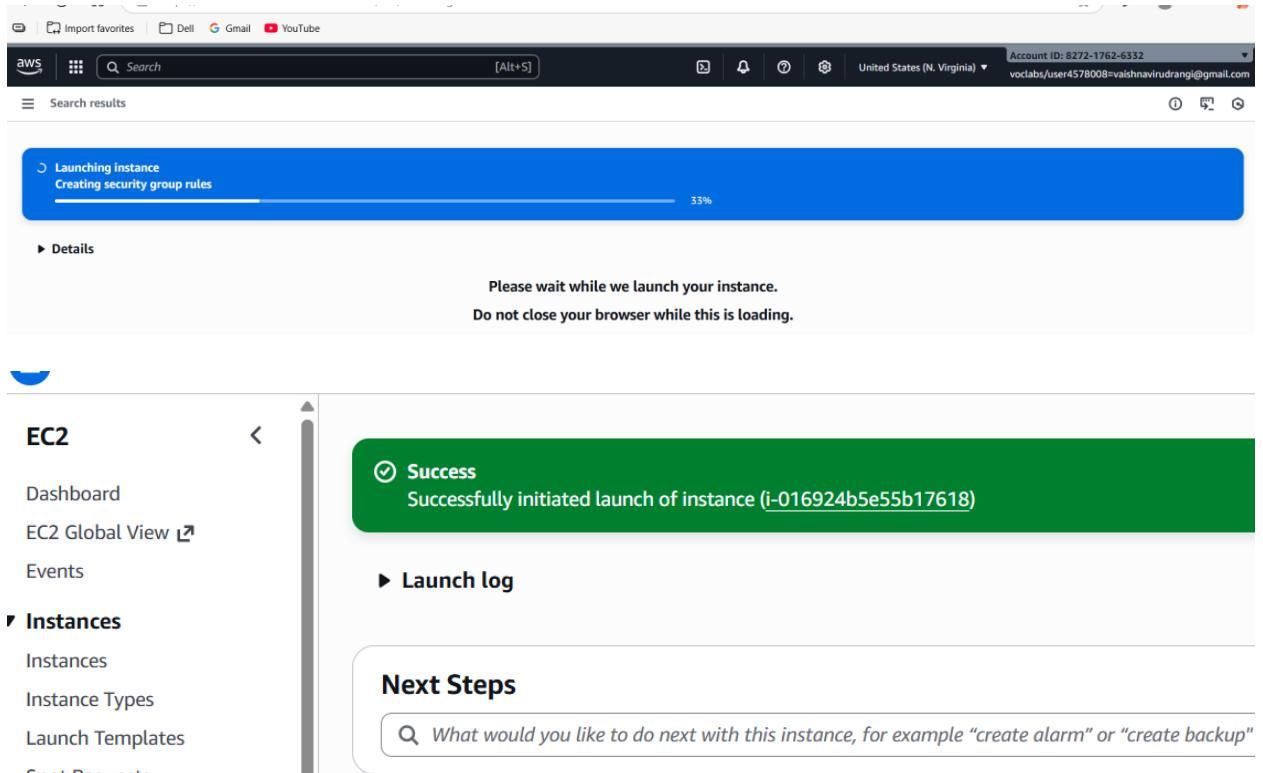
Firewall (security group)
New security group

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

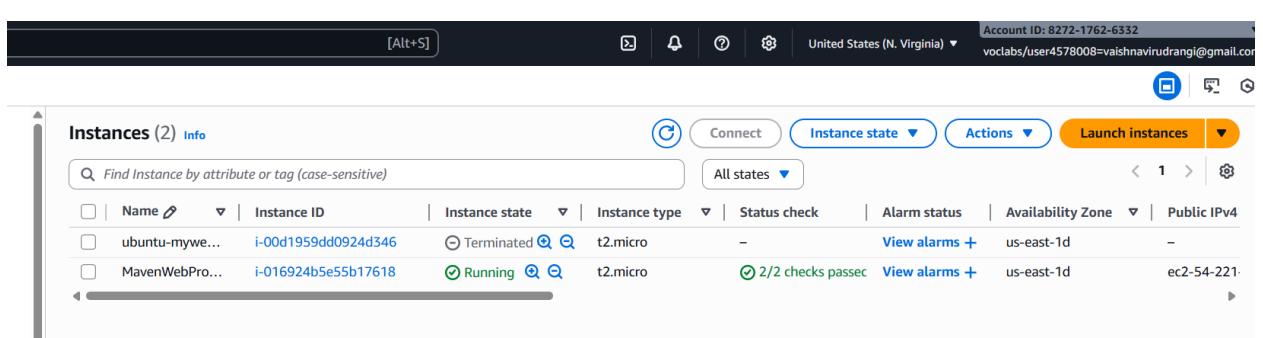
[Cancel](#) [Launch instance](#)

 [Preview code](#)

Stage 9: Number of instances ---1



The screenshot shows the AWS EC2 instance launch process. At the top, a blue progress bar indicates "Launching instance" and "Creating security group rules" at 33%. Below it, a message says "Please wait while we launch your instance. Do not close your browser while this is loading." On the left sidebar, under the "Instances" section, there are links for Instances, Instance Types, Launch Templates, and Spot Requests. In the main area, a green success message box states "Successfully initiated launch of instance (i-016924b5e55b17618)". Below this, a "Launch log" button is visible. A "Next Steps" section with a search bar follows.



The screenshot shows the AWS EC2 Instances page. The title bar includes "Instances (2) Info" and "All states". The main table lists two instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
ubuntu-mywe...	i-00d1959dd0924d346	Terminated	t2.micro	-	View alarms	us-east-1d	-
MavenWebPro...	i-016924b5e55b17618	Running	t2.micro	2/2 checks passed	View alarms	us-east-1d	ec2-54-221-

+++++

Observation - One machines created

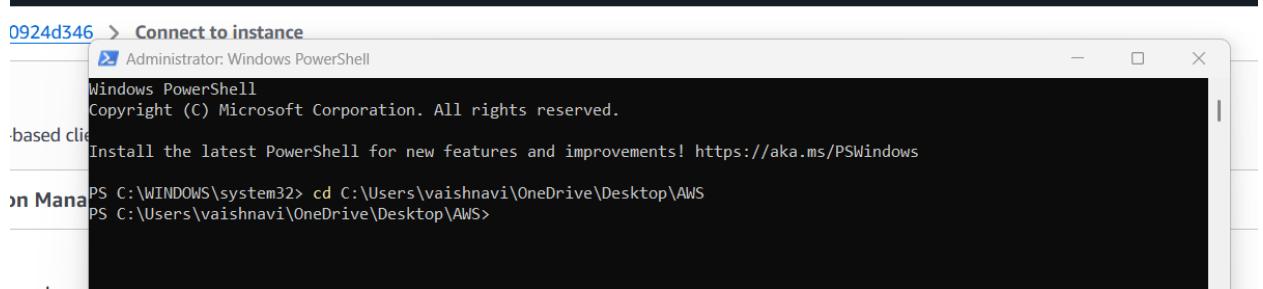
Do this step:---once it is created select that instance and click on connect

Here copy the ssh -i command from SSH client connect tab

The screenshot shows the AWS EC2 Connect interface. At the top, there's a navigation bar with the AWS logo, a search bar, and account information: Account ID: 8272-1762-6332, United States (N. Virginia), and a specific user's email: voclabs/user4578008=vaishnaviudrang@gmail.com. Below the navigation bar, the URL is EC2 > Instances > i-00d1959dd0924d346 > Connect to instance. The main area has tabs: EC2 Instance Connect, Session Manager, **SSH client**, and EC2 serial console. The SSH client tab is selected. Under Instance ID, it shows i-00d1959dd0924d346 (ubuntu-mywebserver). Below this is a list of steps to open an SSH client, including running chmod 400 "my-lab-key.pem". A tooltip says "Command copied" over the chmod command. Another tooltip shows the full command: ssh -i "my-lab-key.pem" ubuntu@ec2-54-175-29-82.compute-1.amazonaws.com. A note at the bottom says: "Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username."

We can use powershell /gitbash /webconsole , to connect to ubuntu machine.

NOTE:- cd path of AWS folder // change path



To connect to above terminals we need to go into the path of the keypair.and

paste the

ssh -i command from the aws console

```

ubuntu@ip-172-31-22-149: ~
PS C:\Users\vaishnavi\OneDrive\Desktop\AWS> ssh -i "ExampleKey.pem" ubuntu@ec2-54-221-108-179.compute-1.amazonaws.com
The authenticity of host 'ec2-54-221-108-179.compute-1.amazonaws.com (54.221.108.179)' can't be established.
ED25519 key fingerprint is SHA256:WBRxDAGR5cG4Mwt9UzhX64DU4UGcNkX52g4sC6ph00s.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-221-108-179.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Thu Nov 13 15:51:46 UTC 2025

  System load: 0.14          Processes:           108
  Usage of /:   25.9% of 6.71GB   Users logged in:      0
  Memory usage: 21%           IPv4 address for enX0: 172.31.22.149
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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

```

IV .Clone the application from github, Write the Dockerfile

once connected to instance

Step 1:- install the docker

install docker ---apt-get update

```

ubuntu@ip-172-31-22-149:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1309 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1585 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [299 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [175 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [15.7 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1498 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [303 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [378 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [31.4 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [2244 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [500 kB]

```

apt-get install docker.io

```
ubuntu@ip-172-31-22-149:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx docker-compose-v2 docker-doc rin
  zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 14 not upgraded.
Need to get 76.0 MB of archives.
After this operation, 288 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 bridge-utils amd64 1.7.1-1ubuntu2 [33.9
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.3.3-0ubuntu1~24.04
]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.28-0ubuntu
38.4 MB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 dns-root-data all 2024071801~ubunt
.1 [5918 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 dnsmasq-base amd64 2.90-2ubuntu
1
```

apt-get install nano

```
ubuntu@ip-172-31-22-149:~$ sudo apt install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.3).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 14 not upgraded.
ubuntu@ip-172-31-22-149:~$ sudo apt install nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nano is already the newest version (7.2-2ubuntu0.1).
nano set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 14 not upgraded.
ubuntu@ip-172-31-22-149:~$
```

step 2:- git clone <paste the github link of maven-web-java project>

```
ubuntu@ip-172-31-22-149:~$ git clone https://github.com/Rudrangi-Vaishnavi/MavenWeb.git
Cloning into 'MavenWeb'...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 26 (delta 1), reused 26 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (26/26), 4.60 KiB | 1.53 MiB/s, done.
Resolving deltas: 100% (1/1), done.
ubuntu@ip-172-31-22-149:~$
```

step 3:- navigate to the maven-web-java project

```
[esolving deltas: 100% (1/1), done.
buntu@ip-172-31-22-149:~$ ls
lavenWeb
buntu@ip-172-31-22-149:~$ cd MavenWeb
buntu@ip-172-31-22-149:~/MavenWeb$ ls
om.xml  src  target
buntu@ip-172-31-22-149:~/MavenWeb$
```

VI. Create the image

Step 4:- nano Dockerfile

```
ubuntu@ip-172-31-22-149:~/MavenWeb
GNU nano 7.2                                     Dockerfile *
FROM tomcat:9-jdk11
COPY target/*.war /usr/local/tomcat/webapps
```

V. Run the image and access it with public ip of virtual machine

Step 1:- build your image

```
docker build -t <imagename> .(dot)
```

```
ubuntu@ip-172-31-22-149:~/MavenWeb$ nano Dockerfile
ubuntu@ip-172-31-22-149:~/MavenWeb$ sudo docker build -t mavenwebproject .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
              Install the buildx component to build images with BuildKit:
              https://docs.docker.com/go/buildx/
Sending build context to Docker daemon 97.79kB
Step 1/2 : FROM tomcat:9-jdk11
9-jdk11: Pulling from library/tomcat
4b3ffd8ccb52: Pulling fs layer
a4b8822e712a: Pulling fs layer
305f2a30fb03: Pulling fs layer
ac56e5323a09: Pulling fs layer
e46c95531a6b: Pulling fs layer
c3224bb24617: Pulling fs layer
4f4fb700ef54: Pulling fs layer
5c56d23d6b20: Pulling fs layer
ac56e5323a09: Waiting
e46c95531a6b: Waiting
c3224bb24617: Waiting
4f4fb700ef54: Waiting
5c56d23d6b20: Waiting
4b3ffd8ccb52: Verifying Checksum
4b3ffd8ccb52: Download complete
ac56e5323a09: Verifying Checksum
ac56e5323a09: Download complete
```

Step 2:- check for images

Step 3:- run image

```
docker run -d --name app-demo -p 6060:8080 <image name>
```

```
--> 179a1b7161ee
Step 2/2 : COPY target/*.war /usr/local/tomcat/webapps
--> ef7ce9986d3a
Successfully built ef7ce9986d3a
Successfully tagged mavenwebproject:latest
ubuntu@ip-172-31-22-149:~/MavenWeb$ sudo docker run -d -p 9090:8080 mavenwebproject
9d68533d1b416ea9e9325d3669d8f929377162deb31857145def0a9a98d5dce7
ubuntu@ip-172-31-22-149:~/MavenWeb$
```

Step:-4 Accessing the app by public ip of virtual machine

Note:-if your are not able to connect change the inbound rules..

The screenshot shows the AWS EC2 Instance Summary page for an instance with ID i-016924b5e55b17618. The instance is named 'ProjectExample'. Key details include:

- Public IPv4 address:** 54.221.108.179 (highlighted with a green border)
- Private IP address:** 172.31.22.149
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-172-31-22-149.ec2.internal
- Instance type:** t2.micro
- VPC ID:** vpc-01281b3c50636b0e4

The screenshot shows a web browser window with the URL 54.221.108.179:9090/MavenWeb/. The page displays the text "Hello World!".

```
ubuntu@ip-172-31-22-149:~/MavenWeb$ sudo docker run -d -p 9090:8080 mavenwebproject
9d68533d1b416ea9e9325d3669d8f929377162deb31857145def0a9a98d5dce7
ubuntu@ip-172-31-22-149:~/MavenWeb$ sudo docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
NAMES
9d68533d1b41 mavenwebproject "catalina.sh run" 8 minutes ago Up 8 minutes 0.0.0.0:9090->8080/tcp, [::]:9090->8080/tcp romantic_euclid
ubuntu@ip-172-31-22-149:~/MavenWeb$ sudo docker stop 9d68533d1b41
9d68533d1b41
ubuntu@ip-172-31-22-149:~/MavenWeb$
```

