Dear Candidate,

Please find the online tasks for the shortlisting process. Read the following instructions carefully and complete the task within a given period of time.

- Complete minimum 10 tasks to get shortlisted for interview round
- This round will test your practical hands on skills w.r.t AWS and DevOps.
- Idea is to test your strengths rather than find your weaknesses. You can try any of the tasks. Need not be in any particular order.
- Once the task is completed kindly upload a final output screenshot after each question.
- Complete the task before 1 PM. No task will be accepted once the time is over.
- Only one task can be submitted by one person, so before submitting please review your task.

How to submit the task:

Step 1: Click on the Online Task Link

Step 2: Download the document

Step 3: Check the online questions

Step 4: Complete the task and take a screenshot for each and every task and upload

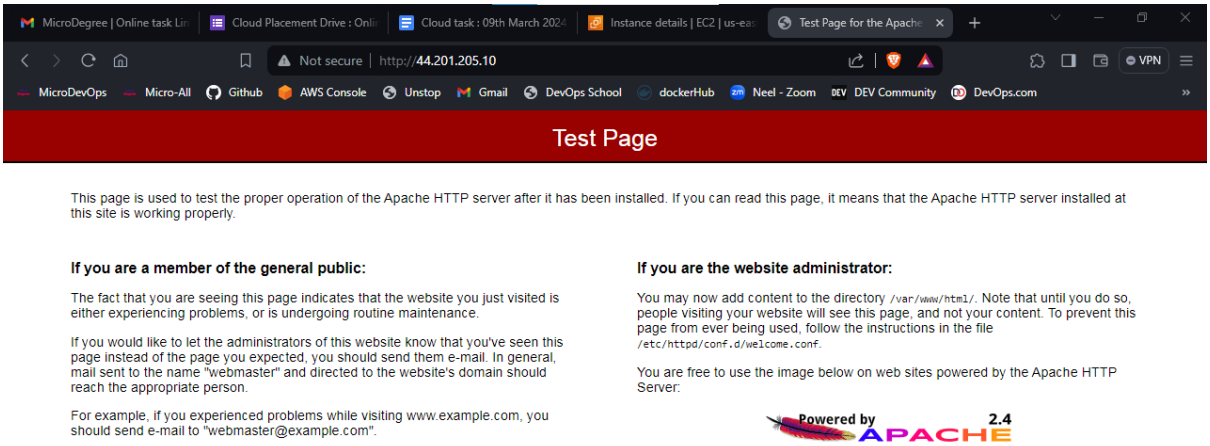
Step 5: Save the document

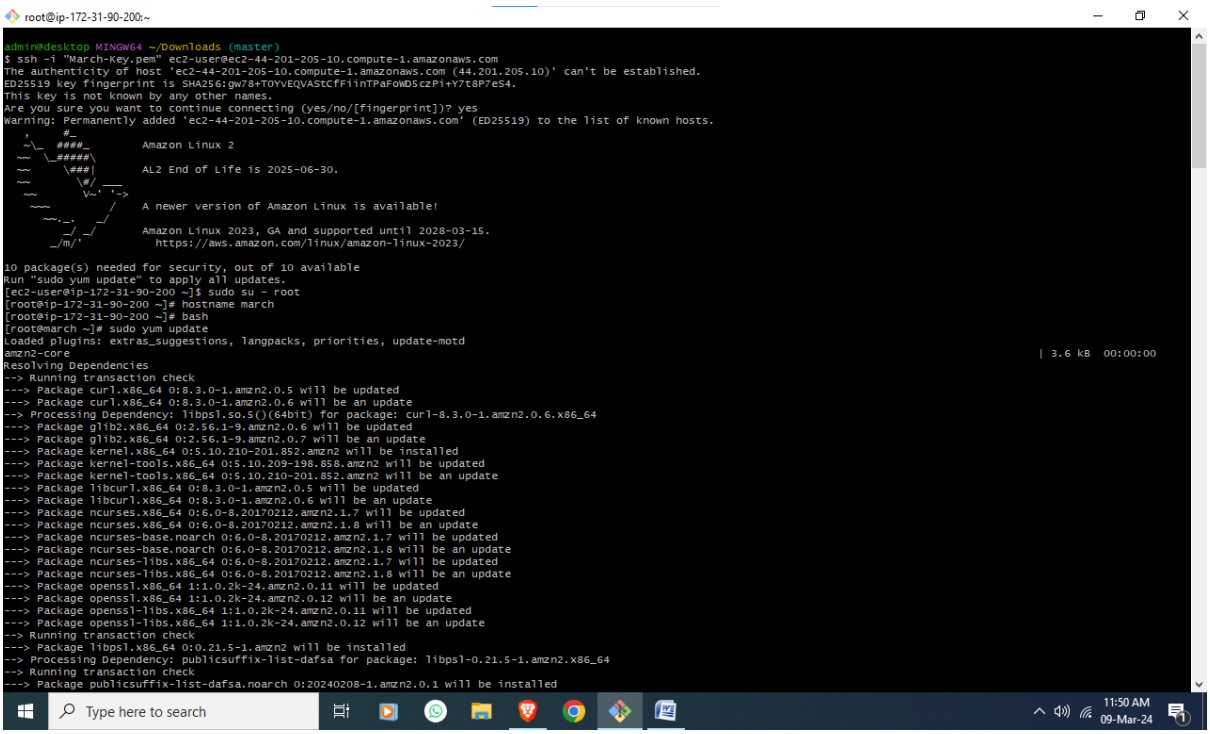
Step 6: Upload the document in Google form

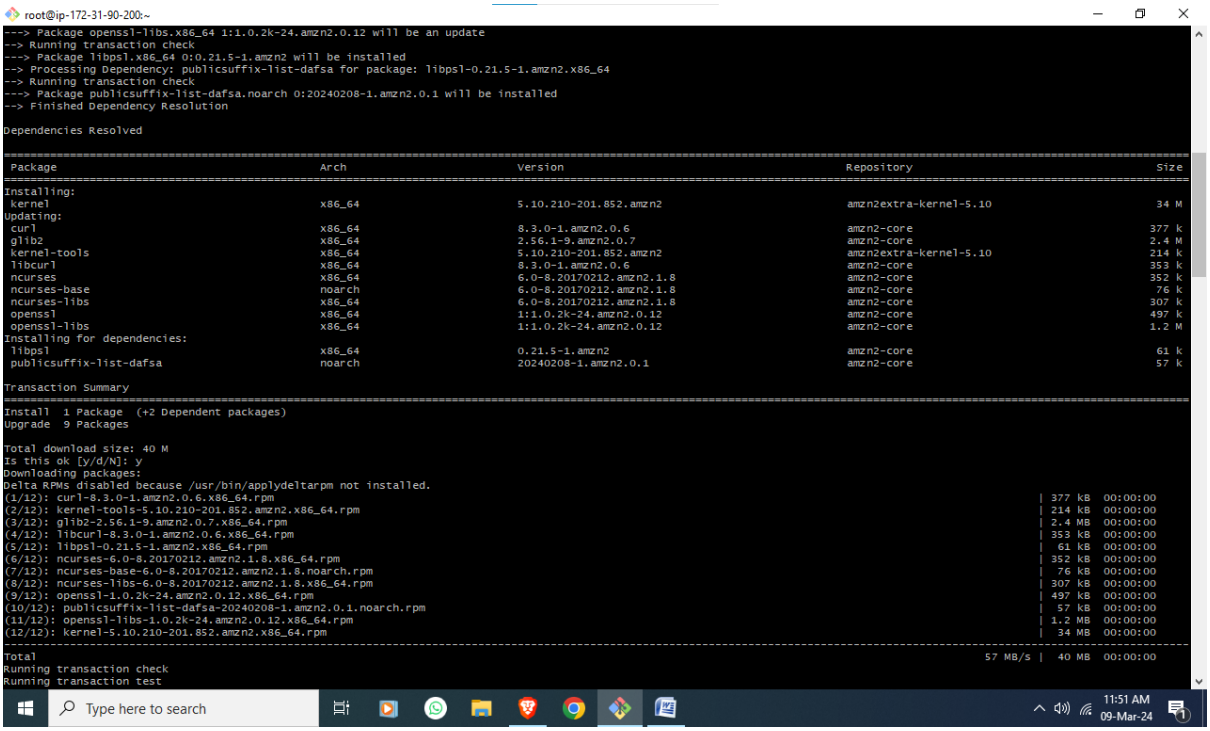
- **AWS Account is mandatory to complete the tasks**.
- **Include AWS account Name in your screenshots**

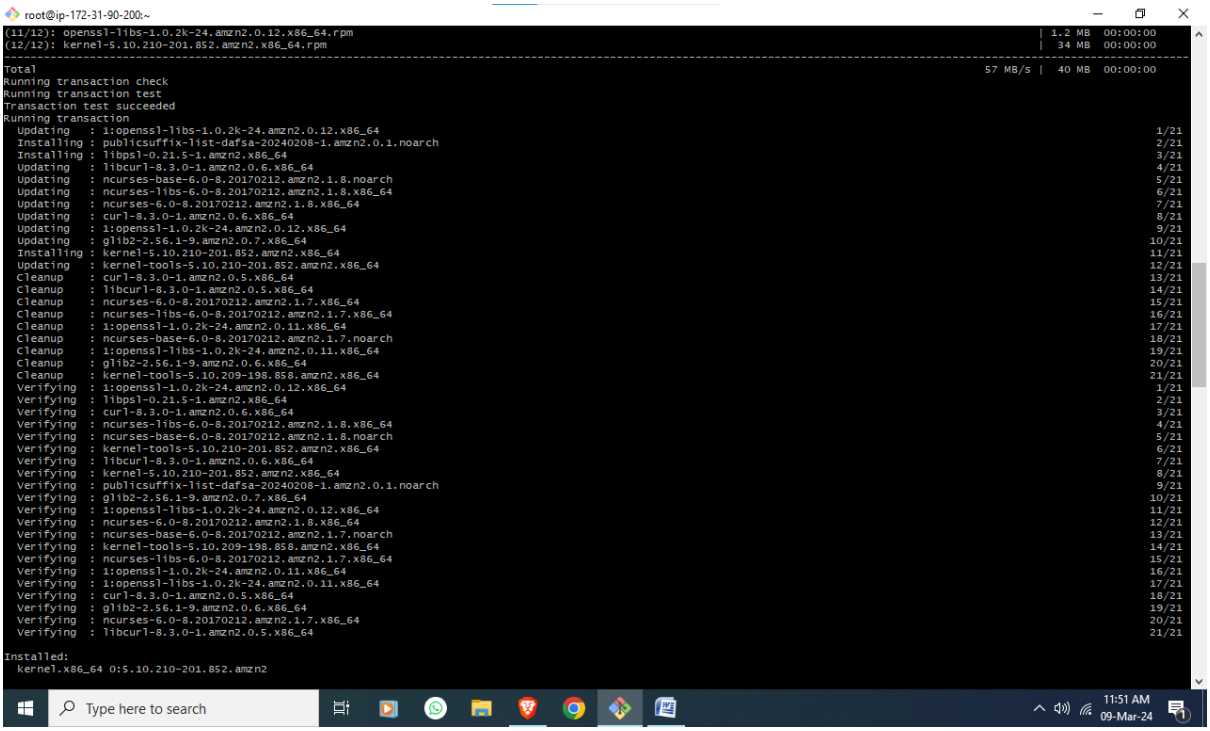
Resume / Copied and similar tasks will be rejected directly.

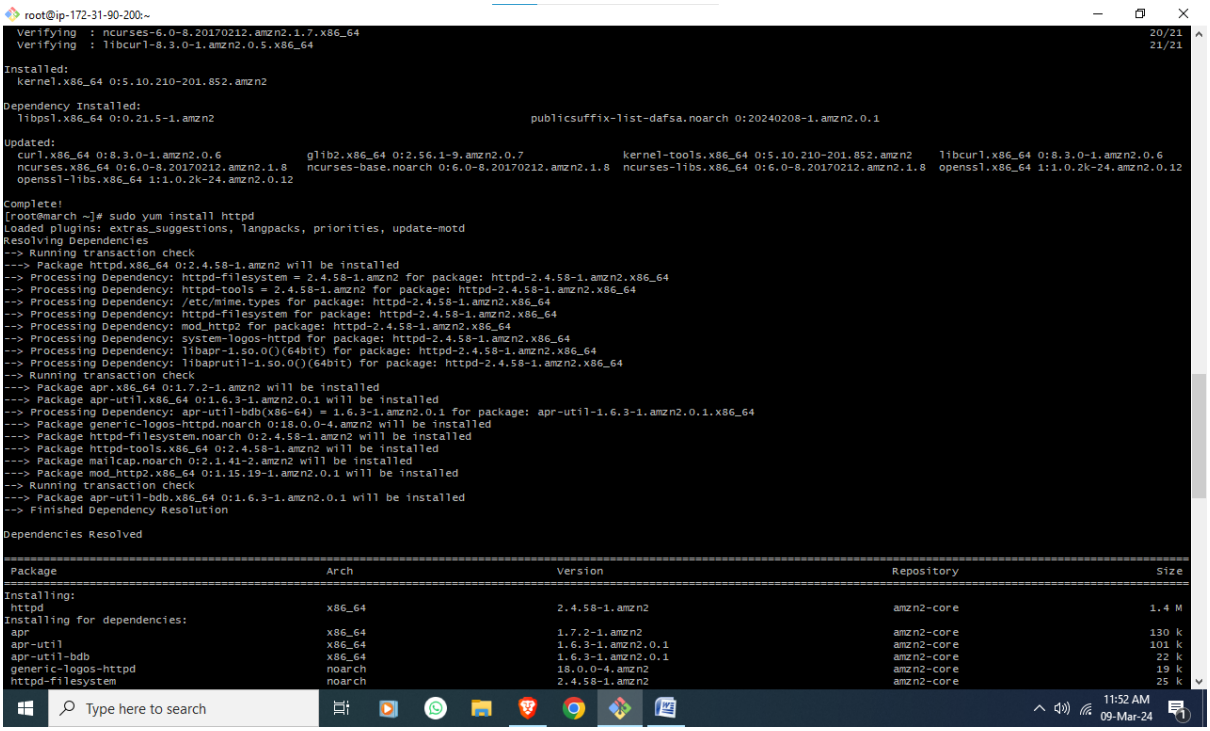
Task 1: Configure ec2 linux machine and install apache configuration
Screenshot:

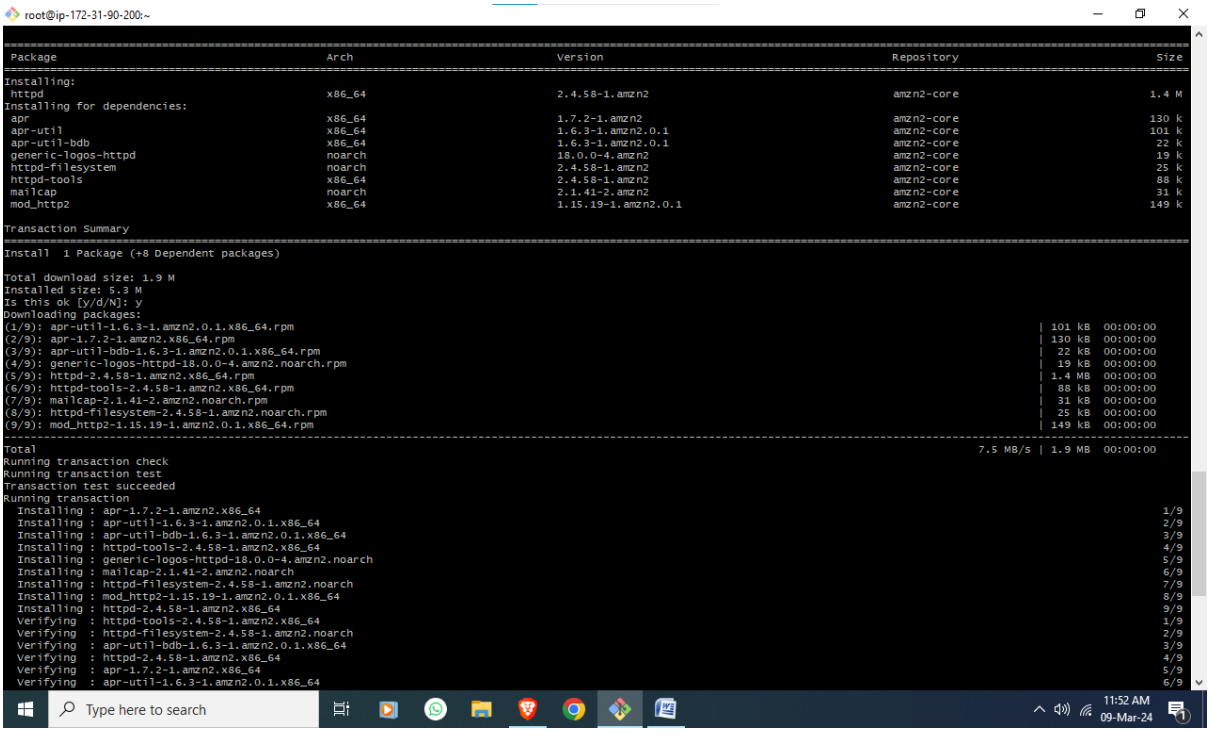





root@ip-172-31-90-200:~
---> Package openssl-libs.x86_64 1:1.0.2k-24.amzn2.0.12 will be an update
--> Running transaction check
---> Package libpsl.x86_64 0:0.21.5-1.amzn2 will be installed
--> Processing Dependency: publicsuffix-list-dafsa for package: libpsl-0.21.5-1.amzn2.x86_64
--> Running transaction check
---> Package publicsuffix-list-dafsa.noarch 0:20240208-1.amzn2.0.1 will be installed
--> Finished Dependency Resolution
Dependencies Resolved
Package Arch Version Repository Size
Installing:
kernel x86_64 5.10.210-201.852.amzn2 amzn2extra-kernel-5.10 34 M
Updating:
curl x86_64 8.3.0-1.amzn2.0.6 amzn2-core 377 k
glib2 x86_64 2.56.1-9.amzn2.0.7 amzn2-core 2.4 M
kernel-tools x86_64 5.10.210-201.852.amzn2 amzn2extra-kernel-5.10 214 k
libcurl x86_64 8.3.0-1.amzn2.0.6 amzn2-core 353 k
ncurses x86_64 6.0-8.20170212.amzn2.1.8 amzn2-core 352 k
ncurses-base noarch 6.0-8.20170212.amzn2.1.8 amzn2-core 76 k
ncurses-libs x86_64 6.0-8.20170212.amzn2.1.8 amzn2-core 307 k
openssl x86_64 1:1.0.2k-24.amzn2.0.12 amzn2-core 497 k
openssl-libs x86_64 1:1.0.2k-24.amzn2.0.12 amzn2-core 1.2 M
Installing for dependencies:
libpsl x86_64 0.21.5-1.amzn2 amzn2-core 61 k
publicsuffix-list-dafsa noarch 20240208-1.amzn2.0.1 amzn2-core 57 k
Transaction Summary
Install 1 Package (+2 Dependent packages)
Upgrade 9 Packages
Total download size: 40 M
Is this ok [y/d/N]: y
Downloading packages:
Delta RPMs disabled because /usr/bin/applydeltarpm not installed.
(1/12): curl-8.3.0-1.amzn2.0.6.x86_64.rpm | 377 kB 00:00:00
(2/12): kernel-tools-5.10.210-201.852.amzn2.x86_64.rpm | 214 kB 00:00:00
(3/12): glib2-2.56.1-9.amzn2.0.7.x86_64.rpm | 2.4 MB 00:00:00
(4/12): libcurl-8.3.0-1.amzn2.0.6.x86_64.rpm | 353 kB 00:00:00
(5/12): libpsl-0.21.5-1.amzn2.x86_64.rpm | 61 kB 00:00:00
(6/12): ncurses-6.0-8.20170212.amzn2.1.8.x86_64.rpm | 352 kB 00:00:00
(7/12): ncurses-base-6.0-8.20170212.amzn2.1.8.noarch.rpm | 76 kB 00:00:00
(8/12): ncurses-libs-6.0-8.20170212.amzn2.1.8.x86_64.rpm | 307 kB 00:00:00
(9/12): openssl-1.0.2k-24.amzn2.0.12.x86_64.rpm | 497 kB 00:00:00
(10/12): publicsuffix-list-dafsa-20240208-1.amzn2.0.1.noarch.rpm | 57 kB 00:00:00
(11/12): openssl-libs-1.0.2k-24.amzn2.0.12.x86_64.rpm | 1.2 MB 00:00:00
(12/12): kernel-5.10.210-201.852.amzn2.x86_64.rpm | 34 MB 00:00:00
Total 57 MB/s | 40 MB 00:00:00
Running transaction check
Running transaction test
Type here to search
11:51 AM
09-Mar-24


root@ip-172-31-90-200:~
(11/12): openssl-libs-1.0.2k-24.amzn2.0.12.x86_64.rpm | 1.2 MB 00:00:00
(12/12): kernel-5.10.210-201.852.amzn2.x86_64.rpm | 34 MB 00:00:00
Total 57 MB/s | 40 MB 00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
Updating : 1:openssl-libs-1.0.2k-24.amzn2.0.12.x86_64 1/21
Installing : publicsuffix-list-dafsa-20240208-1.amzn2.0.1.noarch 2/21
Installing : libpsl-0.21.5-1.amzn2.x86_64 3/21
Updating : libcurl-8.3.0-1.amzn2.0.6.x86_64 4/21
Updating : ncurses-base-6.0-8.20170212.amzn2.1.8.noarch 5/21
Updating : ncurses-libs-6.0-8.20170212.amzn2.1.8.x86_64 6/21
Updating : ncurses-6.0-8.20170212.amzn2.1.8.x86_64 7/21
Updating : curl-8.3.0-1.amzn2.0.6.x86_64 8/21
Updating : 1:openssl-1.0.2k-24.amzn2.0.12.x86_64 9/21
Updating : glib2-2.56.1-9.amzn2.0.7.x86_64 10/21
Installing : kernel-5.10.210-201.852.amzn2.x86_64 11/21
Updating : kernel-tools-5.10.210-201.852.amzn2.x86_64 12/21
Cleanup : curl-8.3.0-1.amzn2.0.5.x86_64 13/21
Cleanup : libcurl-8.3.0-1.amzn2.0.5.x86_64 14/21
Cleanup : ncurses-6.0-8.20170212.amzn2.1.7.x86_64 15/21
Cleanup : ncurses-libs-6.0-8.20170212.amzn2.1.7.x86_64 16/21
Cleanup : 1:openssl-1.0.2k-24.amzn2.0.11.x86_64 17/21
Cleanup : ncurses-base-6.0-8.20170212.amzn2.1.7.noarch 18/21
Cleanup : 1:openssl-libs-1.0.2k-24.amzn2.0.11.x86_64 19/21
Cleanup : glib2-2.56.1-9.amzn2.0.6.x86_64 20/21
Cleanup : kernel-tools-5.10.209-198.858.amzn2.x86_64 21/21
Verifying : 1:openssl-1.0.2k-24.amzn2.0.12.x86_64 1/21
Verifying : libpsl-0.21.5-1.amzn2.x86_64 2/21
Verifying : curl-8.3.0-1.amzn2.0.6.x86_64 3/21
Verifying : ncurses-libs-6.0-8.20170212.amzn2.1.8.x86_64 4/21
Verifying : ncurses-base-6.0-8.20170212.amzn2.1.8.noarch 5/21
Verifying : kernel-tools-5.10.210-201.852.amzn2.x86_64 6/21
Verifying : libcurl-8.3.0-1.amzn2.0.6.x86_64 7/21
Verifying : kernel-5.10.210-201.852.amzn2.x86_64 8/21
Verifying : publicsuffix-list-dafsa-20240208-1.amzn2.0.1.noarch 9/21
Verifying : glib2-2.56.1-9.amzn2.0.7.x86_64 10/21
Verifying : 1:openssl-libs-1.0.2k-24.amzn2.0.12.x86_64 11/21
Verifying : ncurses-6.0-8.20170212.amzn2.1.8.x86_64 12/21
Verifying : ncurses-base-6.0-8.20170212.amzn2.1.7.noarch 13/21
Verifying : kernel-tools-5.10.209-198.858.amzn2.x86_64 14/21
Verifying : ncurses-libs-6.0-8.20170212.amzn2.1.7.x86_64 15/21
Verifying : 1:openssl-1.0.2k-24.amzn2.0.11.x86_64 16/21
Verifying : 1:openssl-libs-1.0.2k-24.amzn2.0.11.x86_64 17/21
Verifying : curl-8.3.0-1.amzn2.0.5.x86_64 18/21
Verifying : glib2-2.56.1-9.amzn2.0.6.x86_64 19/21
Verifying : ncurses-6.0-8.20170212.amzn2.1.7.x86_64 20/21
Verifying : libcurl-8.3.0-1.amzn2.0.5.x86_64 21/21
Installed:
kernel.x86_64 0:5.10.210-201.852.amzn2
Type here to search
11:51 AM
09-Mar-24






```
Installed:
  httpd.x86_64 0:2.4.58-1.amzn2

Dependency Installed:
  apr.x86_64 0:1.7.2-1.amzn2                    apr-util.x86_64 0:1.6.3-1.amzn2.0.1
apr-util-bdb.x86_64 0:1.6.3-1.amzn2.0.1     generic-logos-httpd.noarch 0:18.0.0-
4.amzn2
  httpd-filesystem.noarch 0:2.4.58-1.amzn2    httpd-tools.x86_64 0:2.4.58-1.amzn2
mailcap.noarch 0:2.1.41-2.amzn2               mod_http2.x86_64 0:1.15.19-1.amzn2.0.1

Complete!
```

```
[root@march ~]# sudo service httpd statrt
The service command supports only basic LSB actions (start, stop, restart, try-
restart, reload, force-reload, status). For other actions, please try to use
systemctl.
[root@march ~]# sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
```

Task 2 : Associate an Elastic IP to an EC2 Instance using Terraform

- Create an ec2 instance using terraform workflow

- associate an elastic IP

Screenshot

Insert >> Image >> Upload from computer >> Upload task result screenshot

Task 3 : Implement Auto Scaling: Create an Auto Scaling group that automatically launches new EC2 instances based on predefined rules. You can use the EC2 instance that you created in Task 1 as the base instance for the Auto Scaling group. Test the Auto Scaling group by simulating a surge in traffic to the web server.

Screenshot

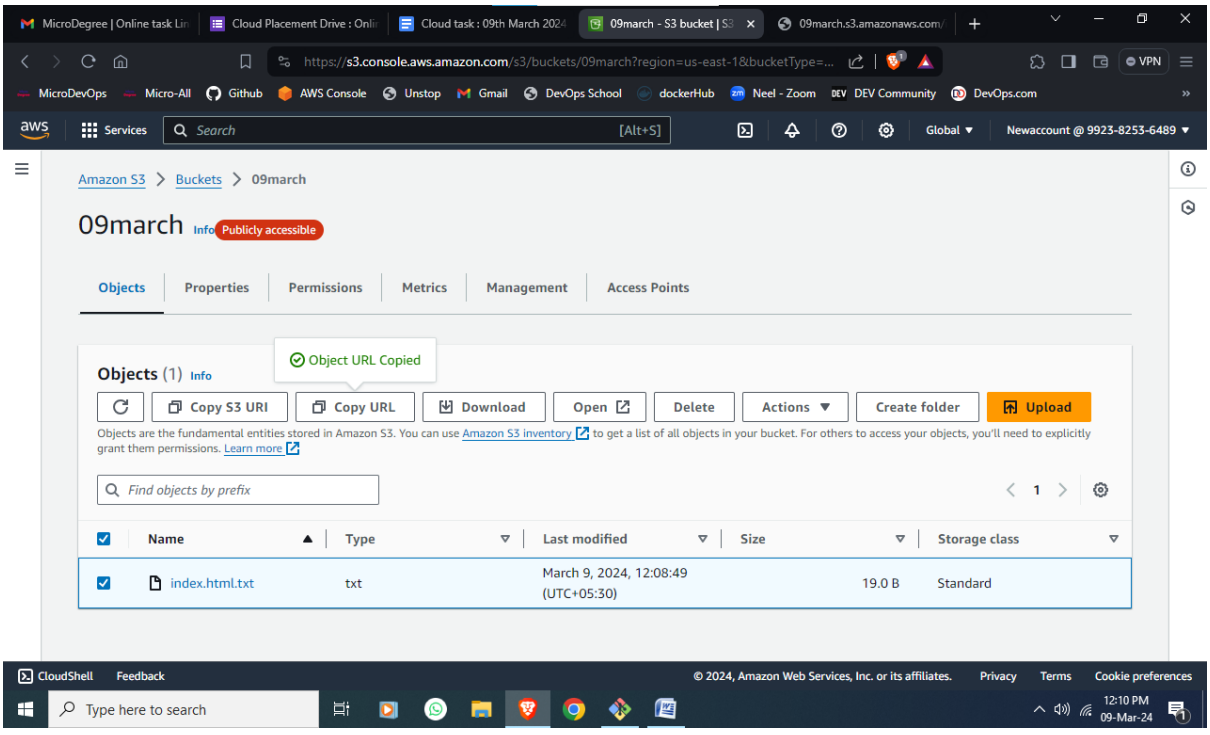
Insert >> Image >> Upload from computer >> Upload task result screenshot

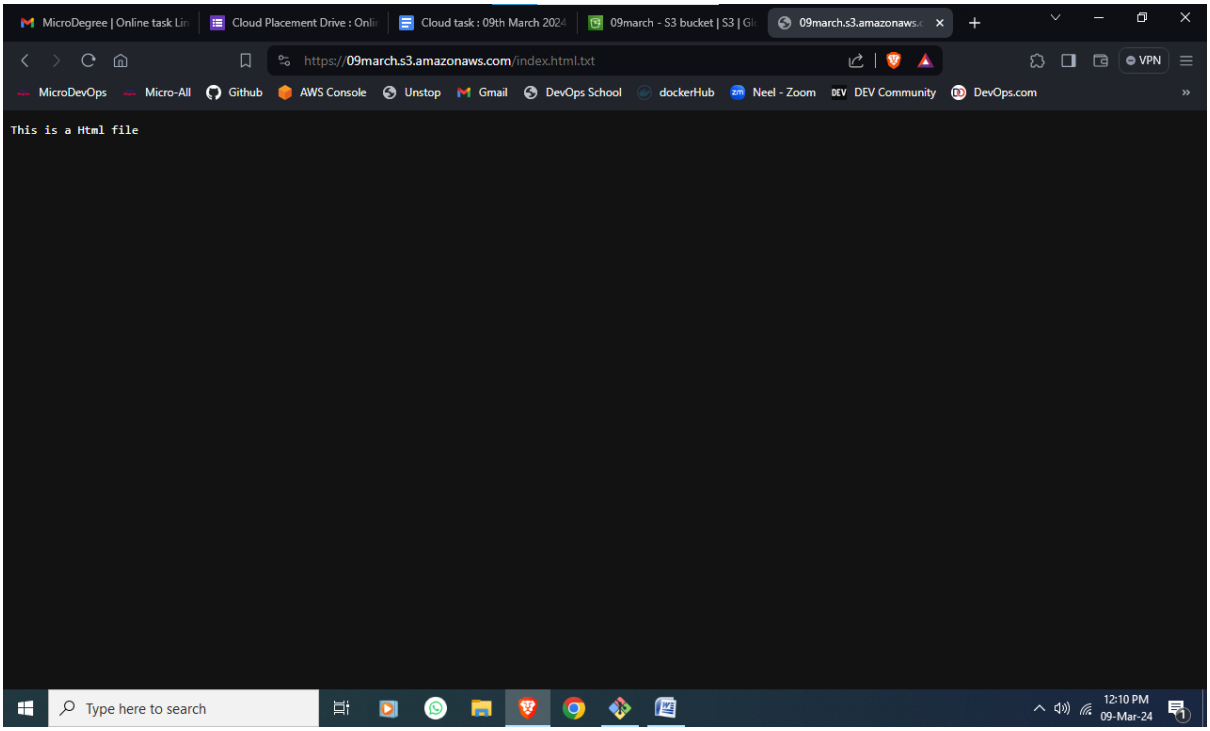
Task 4: Setting Up Continuous Integration and Deployment (CI/CD):
Jenkins is often used for implementing CI/CD pipelines to automate the build, test, and deployment processes.Create a pipeline job using Jenkins Pipeline DSL (declarative or scripted) or a Jenkins file.
Define the stages of your pipeline, including building, testing, code analysis, and deployment.
Configure Jenkins to trigger the pipeline based on code changes, commits, or other

events.

Insert >> Image >> Upload from computer >> Upload task result screenshot

Task 5: Create and configure an S3 bucket: Create a new Amazon Simple Storage Service (S3) bucket and configure it for static website hosting. Then, upload a sample HTML file and ensure that it can be accessed through a web browser.





Task 6 : Host a static website in S3

-> create a s3 bucket with private access ..

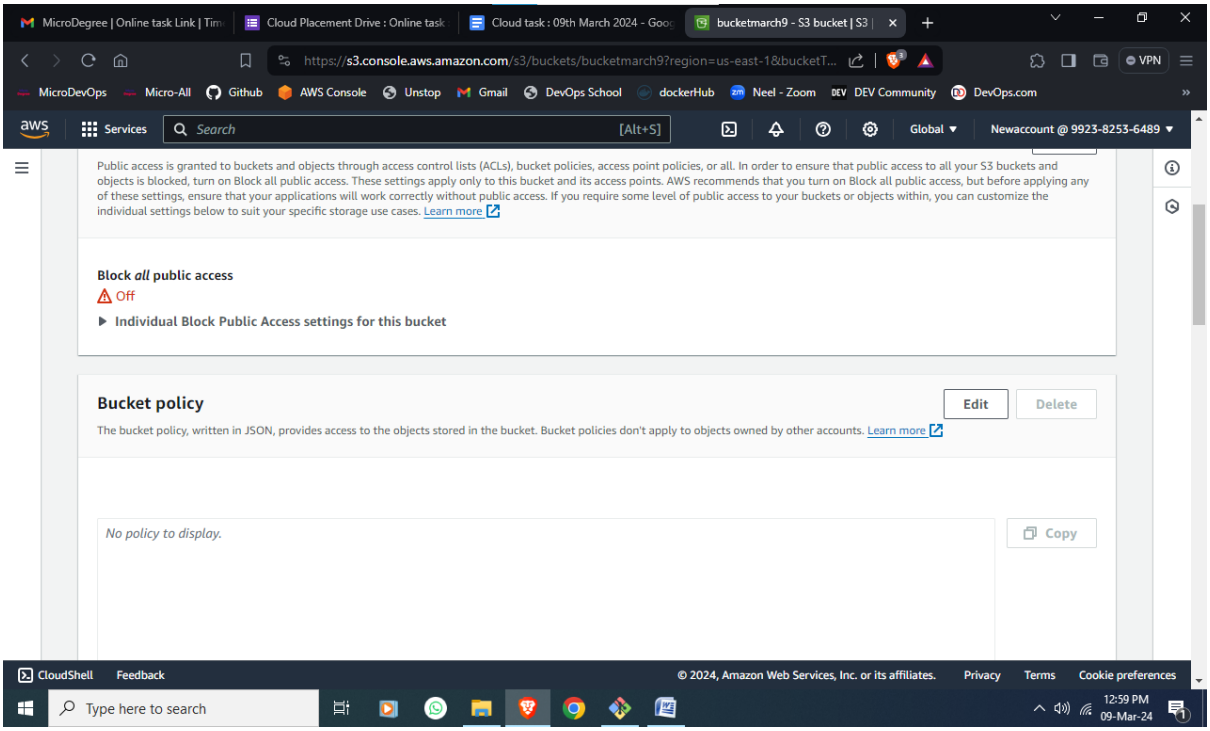
-> Note : bucket should not be Public access

-> ADD the objects -> like front-end code eg: index.html

-> create the Cloud Front (CDN- Content Delivery Network)

-> now you need to Redirect the traffic from Cloud-Front to s3 bucket

-> copy the distribution id of the cloud-front . check in any Browser . website should appear.



Insert >> Image >> Upload from computer >> Upload task result screenshot

Task 7 : Working with Docker Images

- Pull the latest `httpd` image.
- Pull the latest `alpine` image.
- verify images pulled and create 2 containers in each server

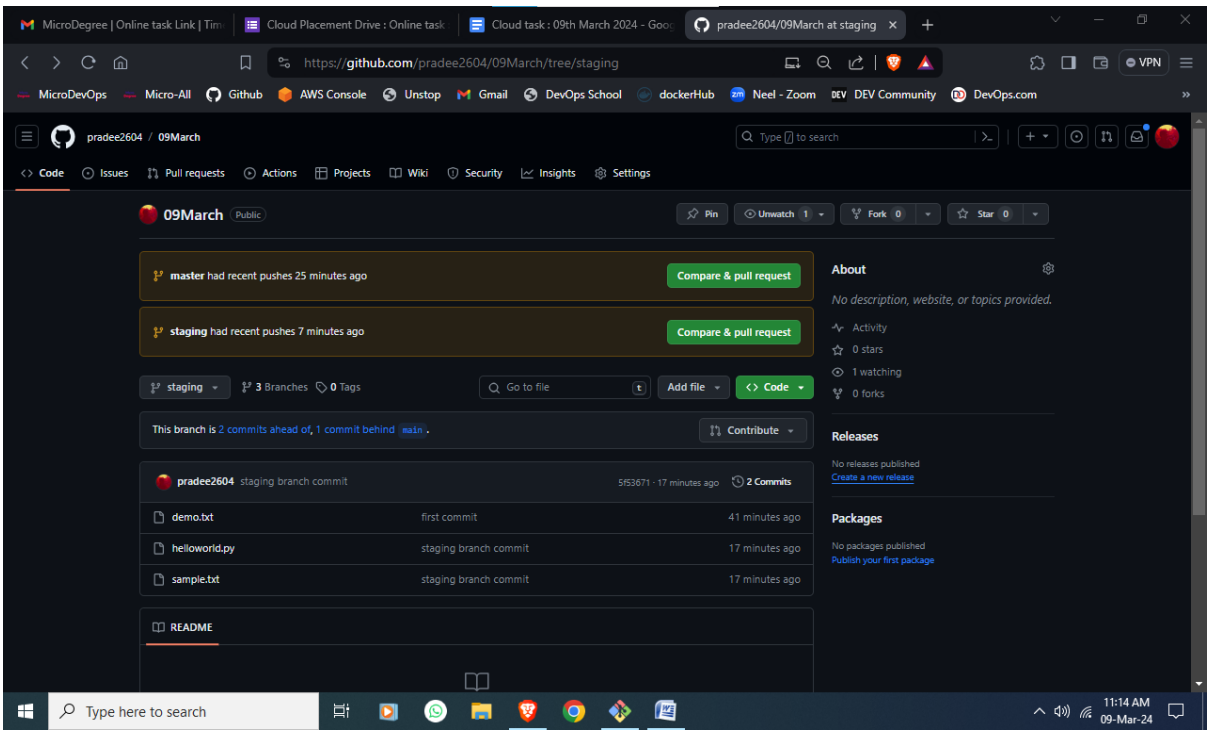
Upload the final output Screenshot

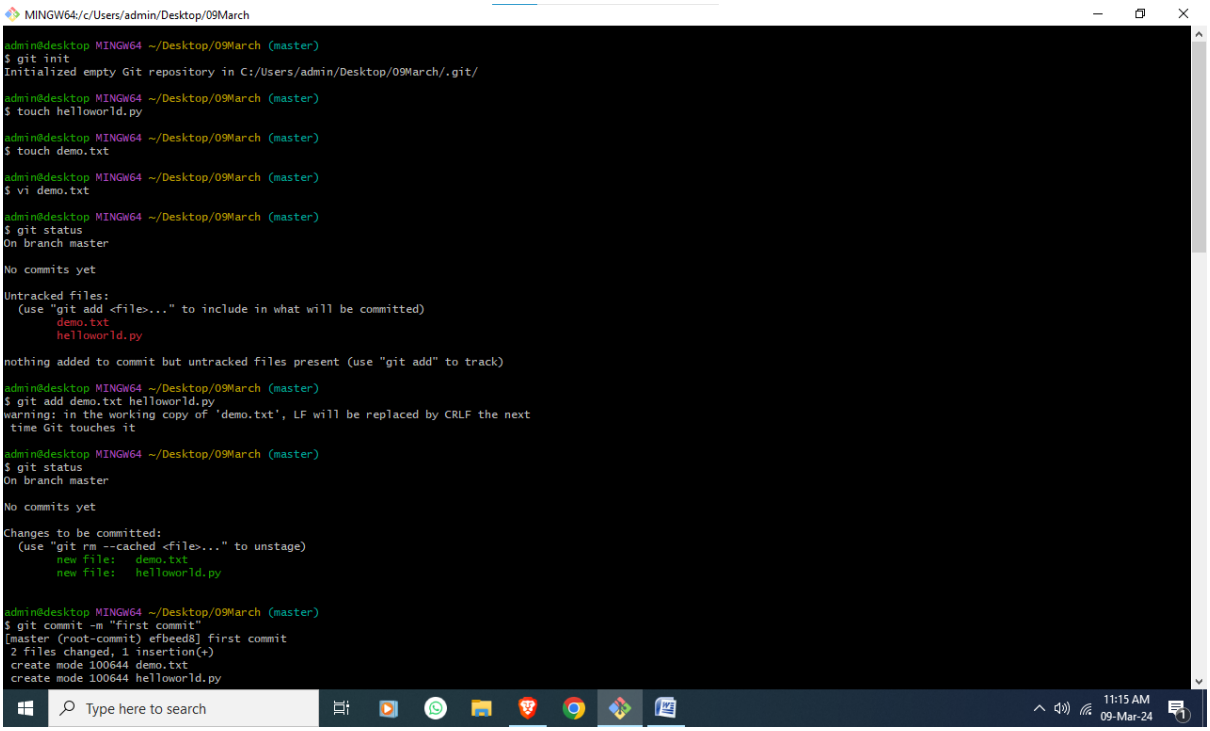
Insert >> Image >> Upload from computer >> Upload task result screenshot

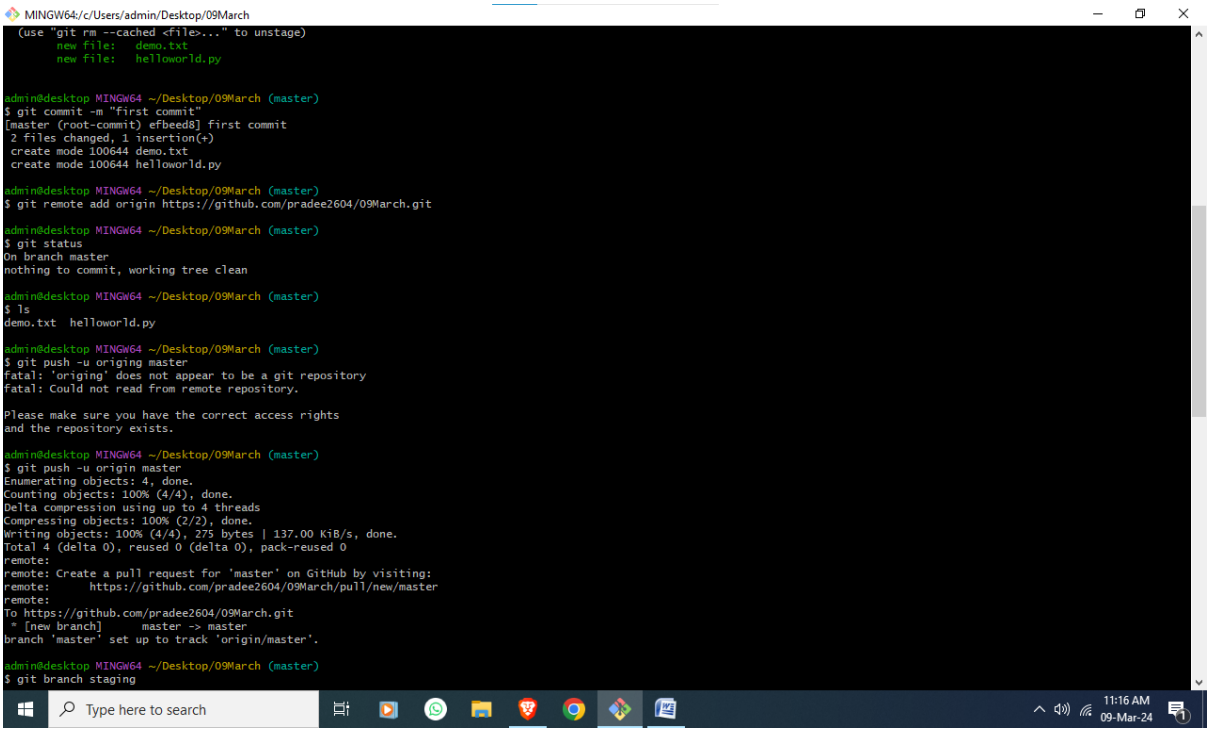
Task : 8 Configure a Load Balancer: Set up an Elastic Load Balancer (ELB) that distributes incoming traffic to the EC2 instances in your Auto Scaling group. Configure health checks to ensure that the ELB only forwards traffic to healthy instances.
Screenshot :

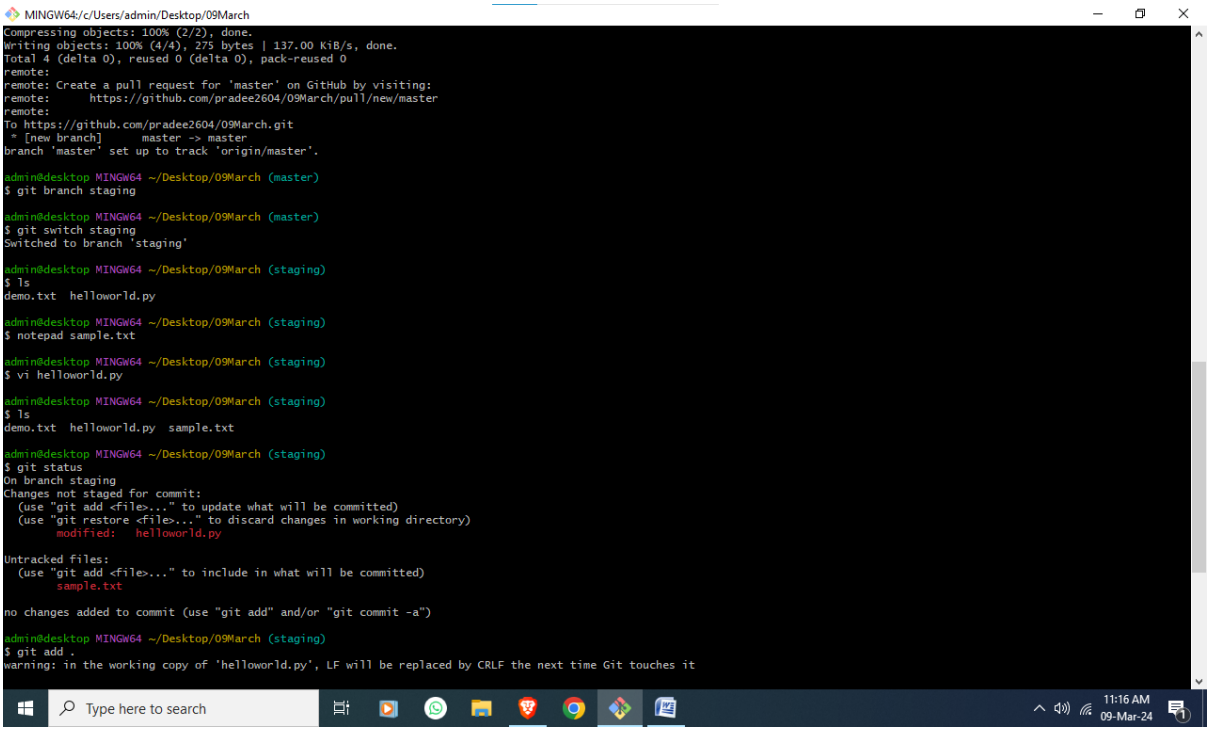
Insert >> Image >> Upload from computer >> Upload task result screenshot

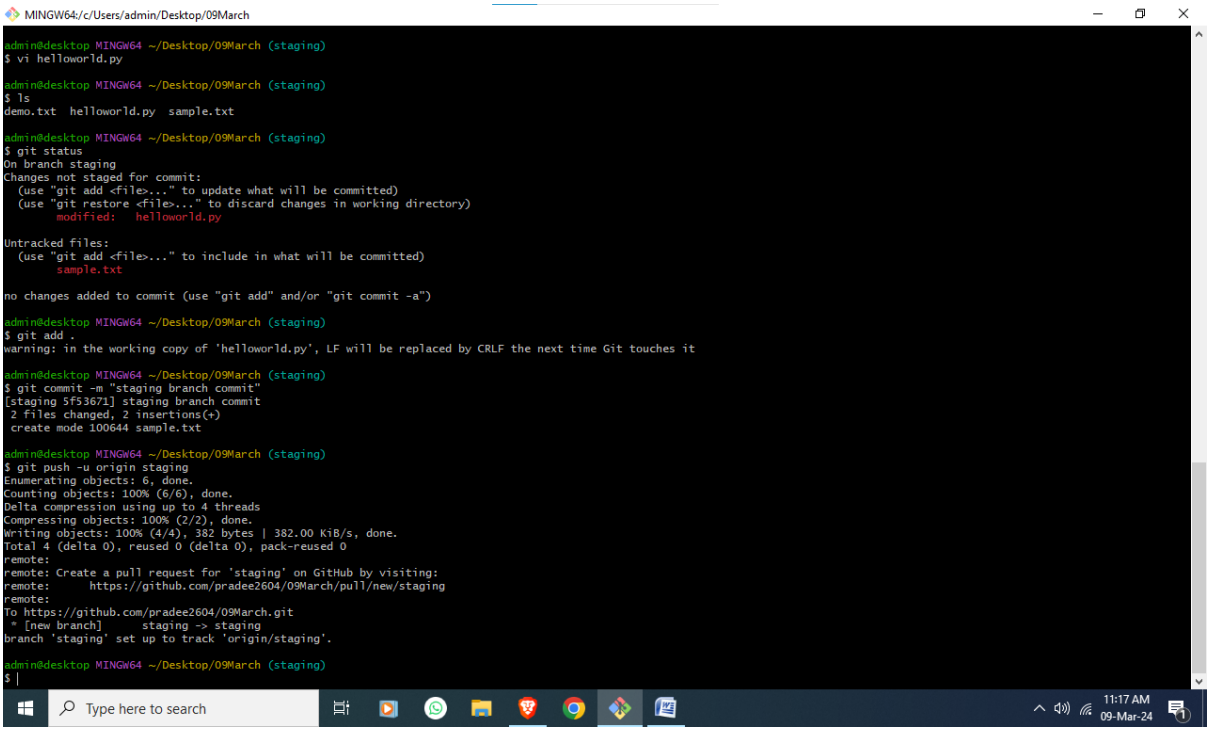
Task 9:
Create a Staging branch in GitHub and push code from the local repository to the Remote and share the full commands screen



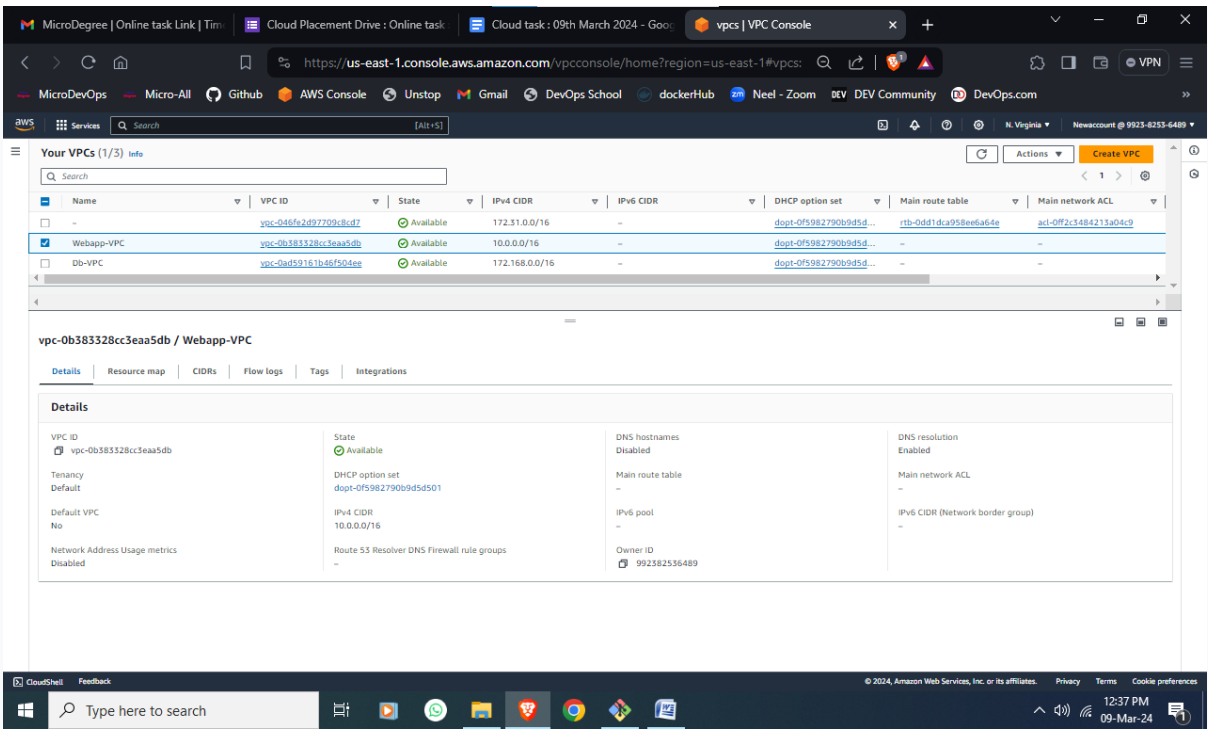

MINGW64:/c/Users/admin/Desktop/09March
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git init
Initialized empty Git repository in C:/Users/admin/Desktop/09March/.git/
admin@desktop MINGW64 ~/Desktop/09March (master)
$ touch helloworld.py
admin@desktop MINGW64 ~/Desktop/09March (master)
$ touch demo.txt
admin@desktop MINGW64 ~/Desktop/09March (master)
$ vi demo.txt
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git status
On branch master
No commits yet
Untracked files:
(use "git add <file>..." to include in what will be committed)
demo.txt
helloworld.py
nothing added to commit but untracked files present (use "git add" to track)
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git add demo.txt helloworld.py
warning: in the working copy of 'demo.txt', LF will be replaced by CRLF the next
time Git touches it
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git status
On branch master
No commits yet
Changes to be committed:
(use "git rm --cached <file>..." to unstage)
new file: demo.txt
new file: helloworld.py
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git commit -m "first commit"
[master (root-commit) efbeed8] first commit
2 files changed, 1 insertion(+)
create mode 100644 demo.txt
create mode 100644 helloworld.py
Type here to search
11:15 AM
09-Mar-24

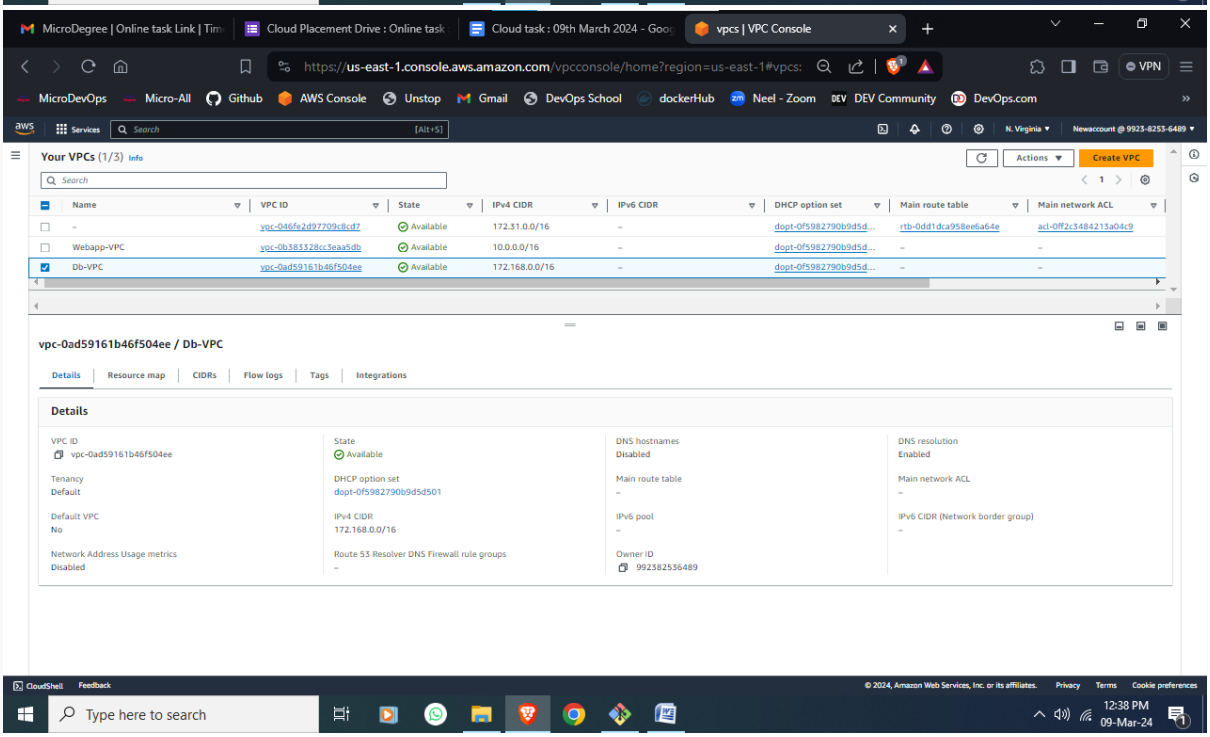

MINGW64:/c/Users/admin/Desktop/09March
(use "git rm --cached <file>..." to unstage)
new file: demo.txt
new file: helloworld.py
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git commit -m "first commit"
[master (root-commit) efbeed8] first commit
2 files changed, 1 insertion(+)
create mode 100644 demo.txt
create mode 100644 helloworld.py
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git remote add origin https://github.com/pradee2604/09March.git
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git status
On branch master
nothing to commit, working tree clean
admin@desktop MINGW64 ~/Desktop/09March (master)
$ ls
demo.txt helloworld.py
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git push -u origing master
fatal: 'origing' does not appear to be a git repository
fatal: Could not read from remote repository.
Please make sure you have the correct access rights
and the repository exists.
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git push -u origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 275 bytes | 137.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote: https://github.com/pradee2604/09March/pull/new/master
remote:
To https://github.com/pradee2604/09March.git
* [new branch] master -> master
branch 'master' set up to track 'origin/master'.
admin@desktop MINGW64 ~/Desktop/09March (master)
$ git branch staging
Type here to search
11:16 AM
09-Mar-24

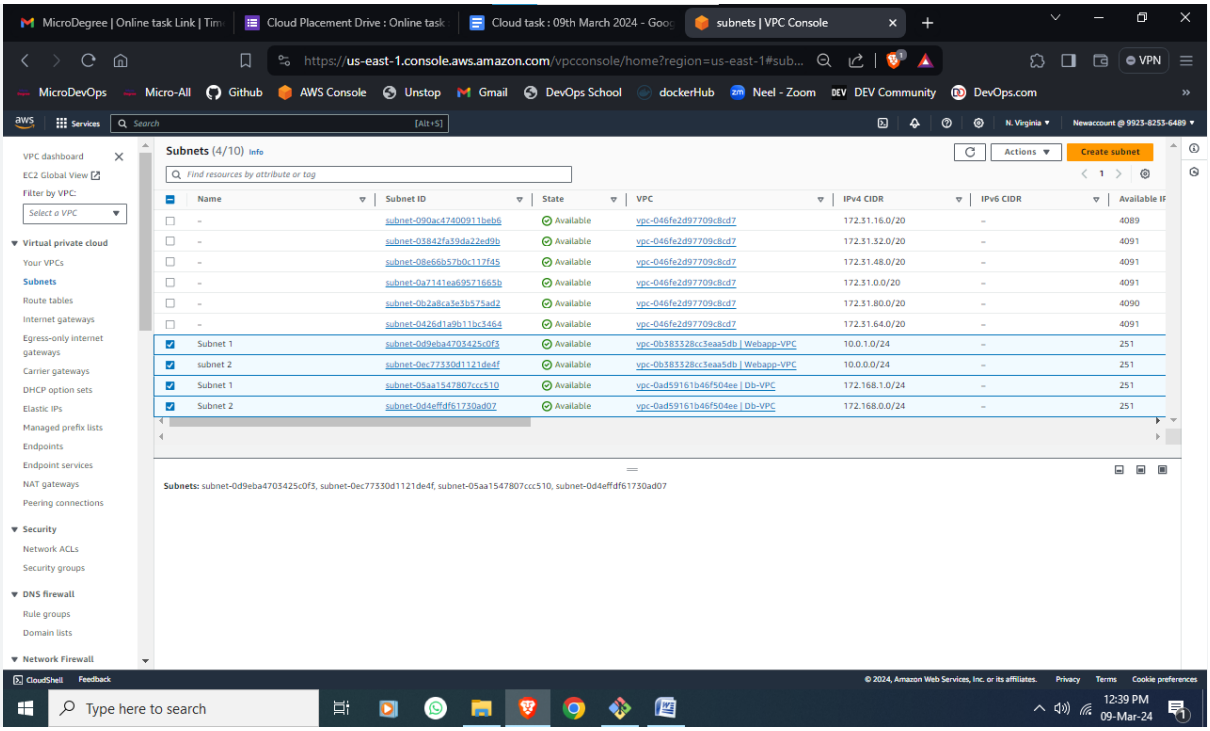




Task 10: Create 2 VPC's Named "Webapp-VPC" & "Db-VPC"
It should have 2 Subnets each, one with Class A IPv4 CIDR and Class B IPv4 CIDR, and 255 ports in each subnet.


MicroDegree | Online task Link | Tim
Cloud Placement Drive : Online task
Cloud task : 09th March 2024 - Goo
vpcs | VPC Console
https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#vpcs:
VPN
MicroDevOps
Micro-All
Github
AWS Console
Unstop
Gmail
DevOps School
dockerHub
Neel - Zoom
DEV Community
DevOps.com
Services
Search
[Alt+S]
N. Virginia
Newaccount @ 9923-8253-6489
Your VPCs (1/3) Info
Actions
Create VPC
Name
VPC ID
State
IPv4 CIDR
IPv6 CIDR
DHCP option set
Main route table
Main network ACL
vpc-046fe2d97709c8cd7
Available
172.31.0.0/16
dopt-0f5982790b9d5d...
rtb-0dd1dca958ee6a64e
acl-0ff2c3484213a04c9
Webapp-VPC
vpc-0b383328cc3eaa5db
Available
10.0.0.0/16
dopt-0f5982790b9d5d...
Db-VPC
vpc-0ad59161b46f504ee
Available
172.168.0.0/16
dopt-0f5982790b9d5d...
vpc-0b383328cc3eaa5db / Webapp-VPC
Details
Resource map
CIDRs
Flow logs
Tags
Integrations
Details
VPC ID
vpc-0b383328cc3eaa5db
State
Available
DNS hostnames
Disabled
DNS resolution
Enabled
Tenancy
Default
DHCP option set
dopt-0f5982790b9d5d501
Main route table
Main network ACL
Default VPC
No
IPv4 CIDR
10.0.0.0/16
IPv6 pool
IPv6 CIDR (Network border group)
Network Address Usage metrics
Disabled
Route 53 Resolver DNS Firewall rule groups
Owner ID
992382536489
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Type here to search
12:37 PM
09-Mar-24


MicroDegree | Online task Link | Tim
Cloud Placement Drive : Online task
Cloud task : 09th March 2024 - Goo
vpcs | VPC Console
https://us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#vpcs:
VPN
MicroDevOps
Micro-All
Github
AWS Console
Unstop
Gmail
DevOps School
dockerHub
Neel - Zoom
DEV Community
DevOps.com
Services
Search
[Alt+S]
N. Virginia
Newaccount @ 9923-8253-6489
Your VPCs (1/3) Info
Actions
Create VPC
Name
VPC ID
State
IPv4 CIDR
IPv6 CIDR
DHCP option set
Main route table
Main network ACL
vpc-046fe2d97709c8cd7
Available
172.31.0.0/16
dopt-0f5982790b9d5d...
rtb-0dd1dca958ee6a64e
acl-0ff2c3484213a04c9
Webapp-VPC
vpc-0b383328cc3eaa5db
Available
10.0.0.0/16
dopt-0f5982790b9d5d...
Db-VPC
vpc-0ad59161b46f504ee
Available
172.168.0.0/16
dopt-0f5982790b9d5d...
vpc-0ad59161b46f504ee / Db-VPC
Details
Resource map
CIDRs
Flow logs
Tags
Integrations
Details
VPC ID
vpc-0ad59161b46f504ee
State
Available
DNS hostnames
Disabled
DNS resolution
Enabled
Tenancy
Default
DHCP option set
dopt-0f5982790b9d5d501
Main route table
Main network ACL
Default VPC
No
IPv4 CIDR
172.168.0.0/16
IPv6 pool
IPv6 CIDR (Network border group)
Network Address Usage metrics
Disabled
Route 53 Resolver DNS Firewall rule groups
Owner ID
992382536489
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Type here to search
12:38 PM
09-Mar-24



Task 11: Build a custom docker image using Ubuntu as a base docker image and run the nginx application. - this docker image should be built using a Dockerfile. Once the docker image is build, start the docker image using the host network and make it accessible on Public IP

Screenshot:

Insert >> Image >> Upload from computer >> Upload task result screenshot

Task 12: Create a basic deployment pipeline that deploys your application to different environments, such as development, staging, and production. Implement testing and approval steps for each environment.

Screenshot:

Insert >> Image >> Upload from computer >> Upload task result screenshot

Scenario based questions:

Task 13 :Scenario: You have a running process that is consuming a lot of system resources and needs to be terminated. How would you find the process ID (PID) of the process and terminate it gracefully?

Explain in 300 words

Task 14: Scenario: You are using a computer with a single-core processor. You have multiple applications open, including a web browser, a text editor, and a media player. The media player freezes and becomes unresponsive. Explain the potential reasons behind this issue and suggest a possible solution.

Single processor will not work if the load is more . it will work very slowly and sometimes it will not work. it's like playing a big game in a small laptops which has only basic features.
so we should have proper resource else we should make the present machine by upgrading.
in worst condition we can use the application by closing all other applications and providing all necessary things which are required for the application to run.

also we can run the application using virtual machines and using the image file which is available in docker container

with these steps somehow we can run the application
still upgrading the machine or high-end machines performance is still better.

Task15: A user reports that they are unable to access a website by its domain name, but other websites are working fine. What steps would you take to troubleshoot and resolve this issue from the DNS perspective?

Explain in 300 words