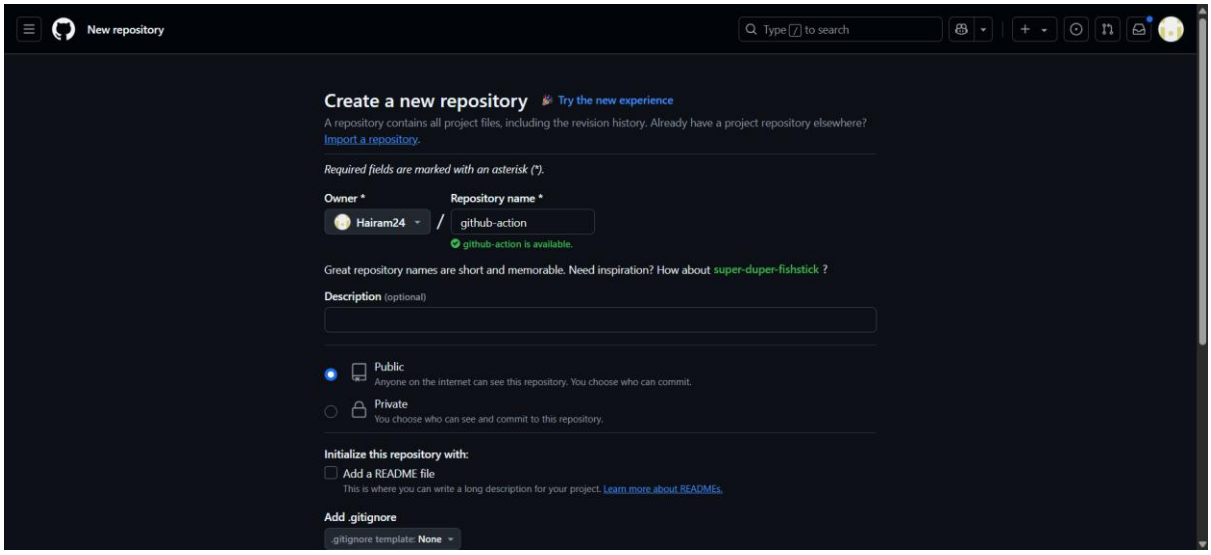
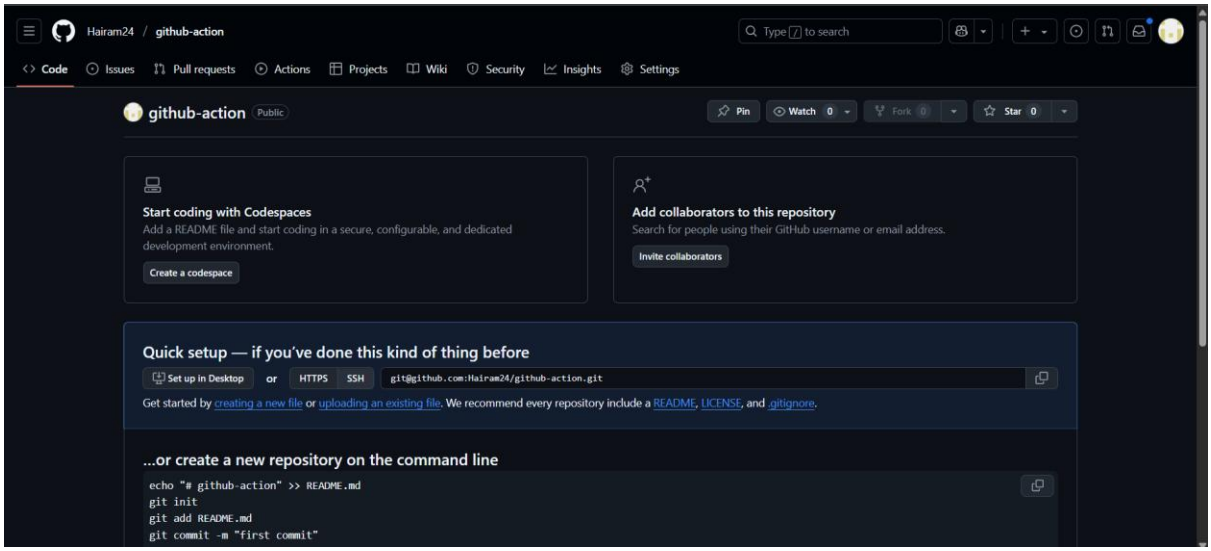
1) Easy

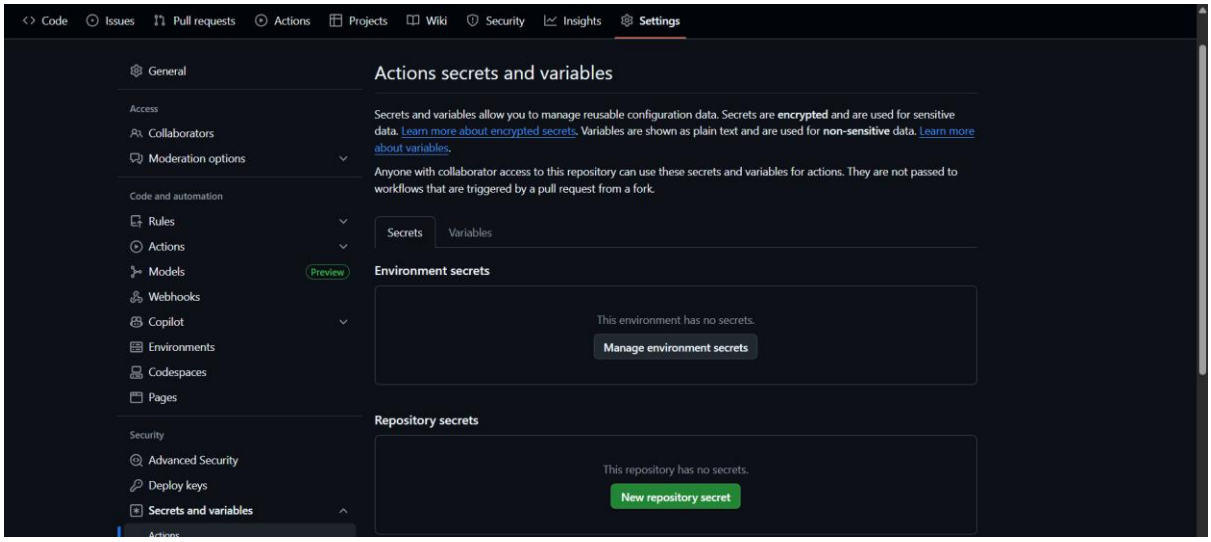
Using CI/CD tool GitHub actions deploy a static web site on AWS S3. And website should be expose publicly



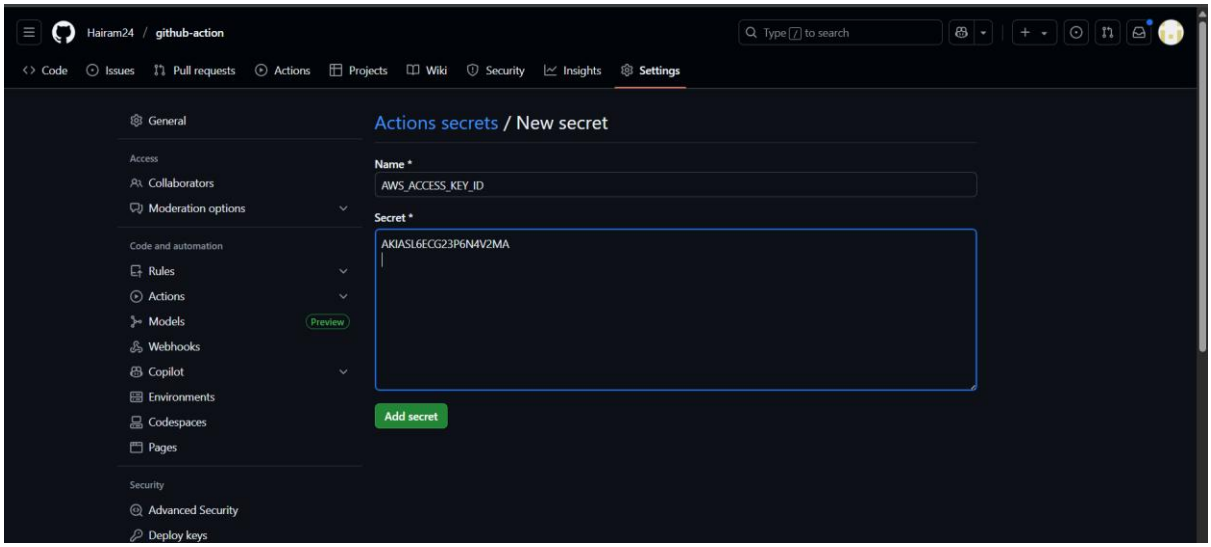
New public repo is created.



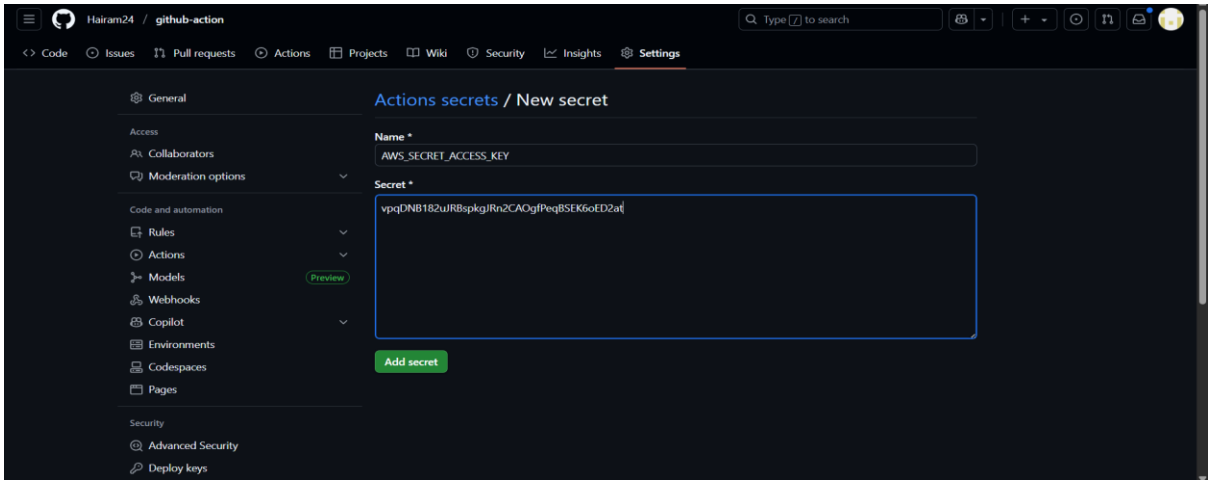
Repository successfully created.



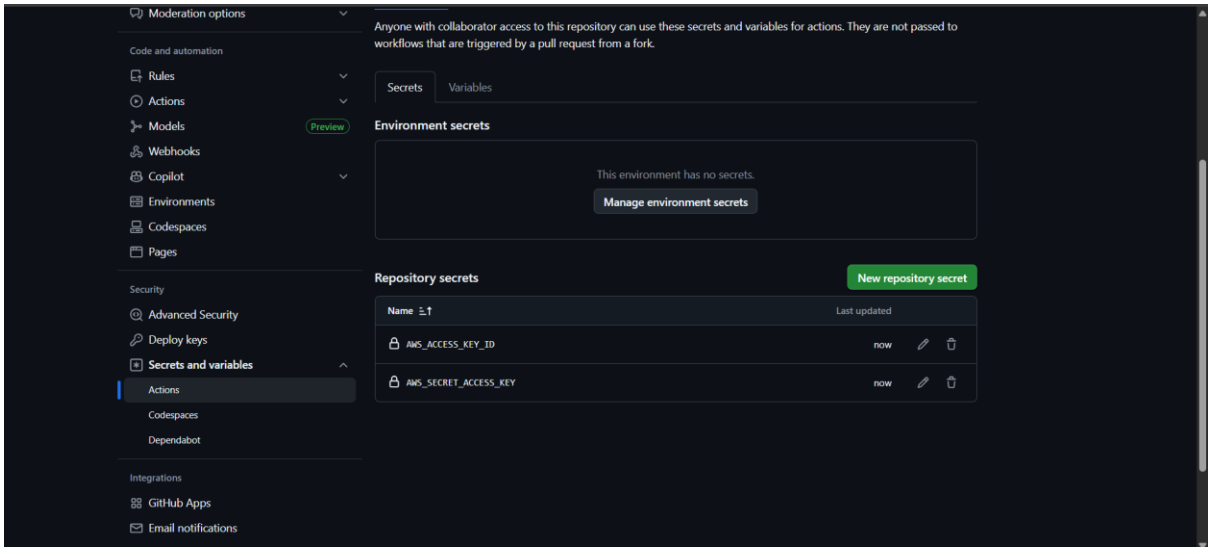
To add keys > secrets and variables> actions.



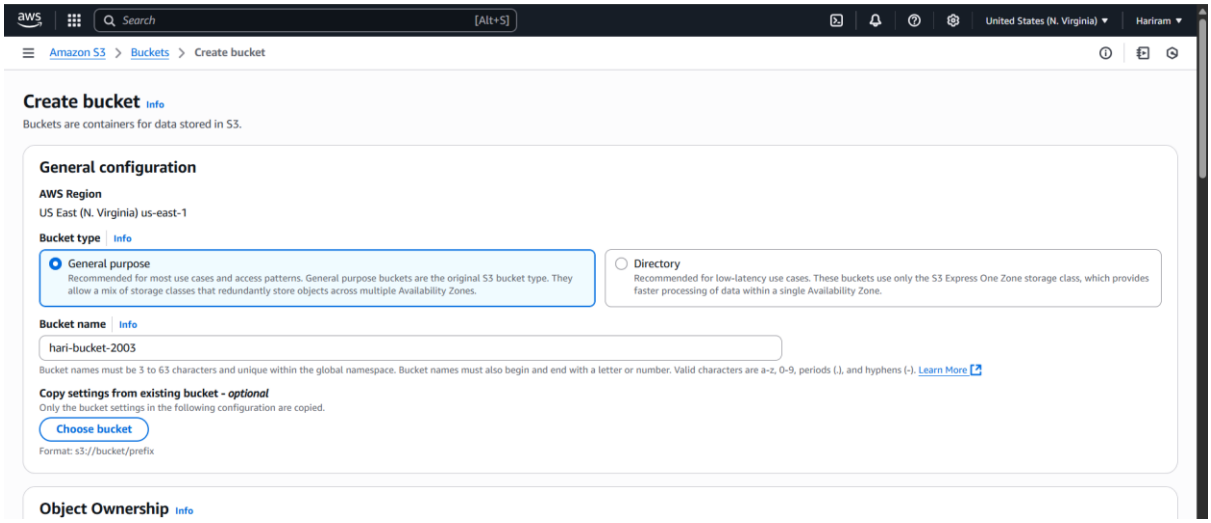
Added the aws_acces_key



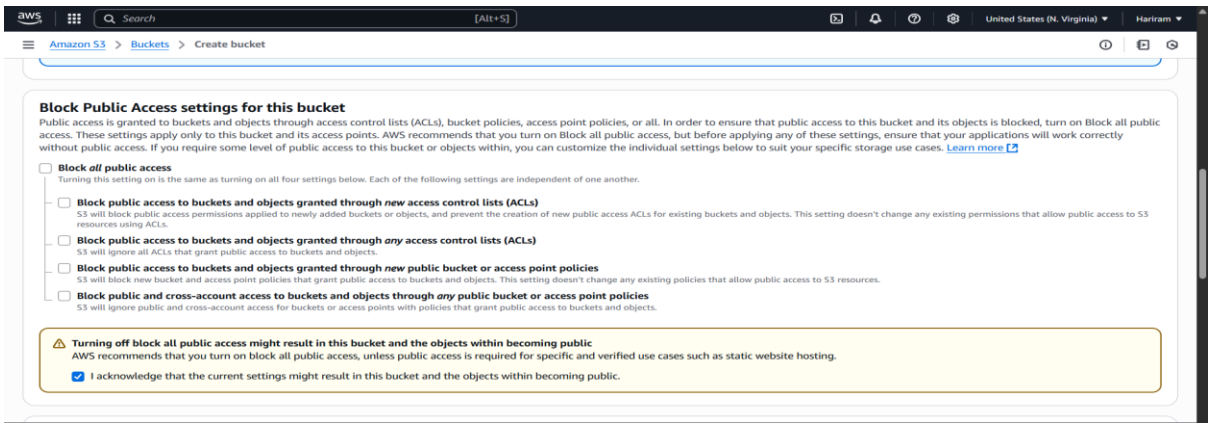
Added the aws_secret_access_key.



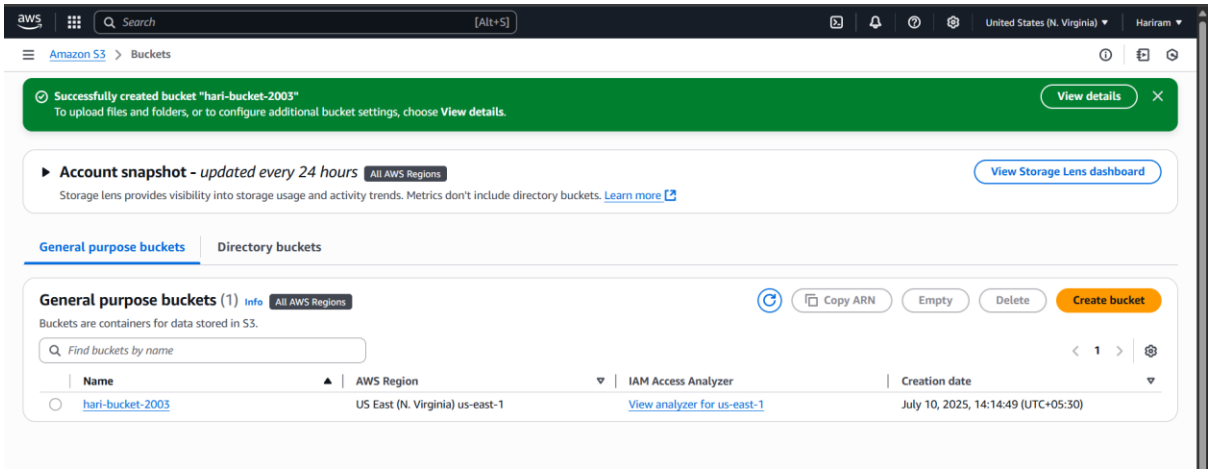
Both keys are added.



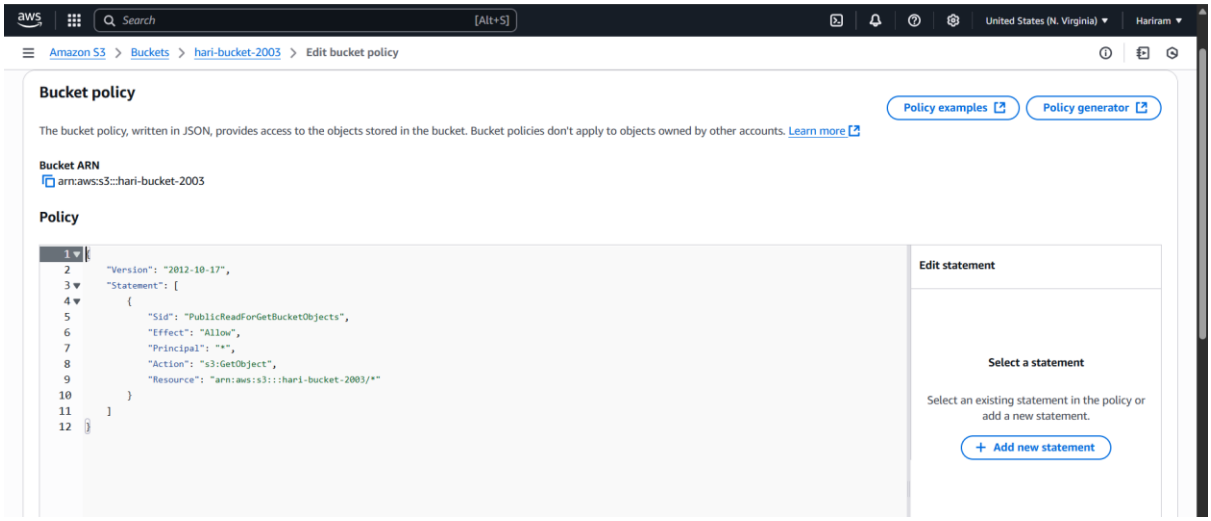
New S3 bucket is created .



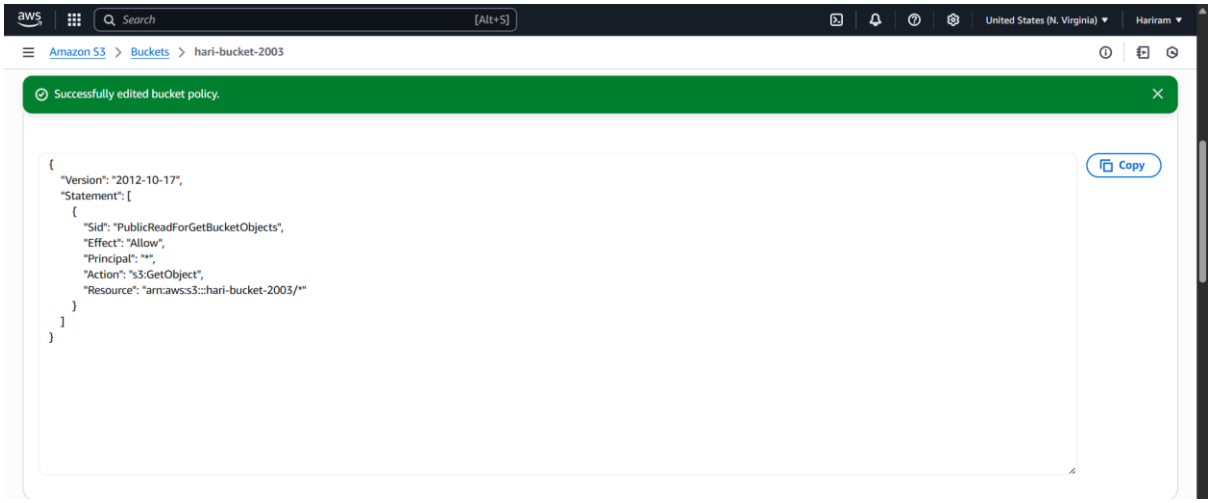
Block public access is disables.



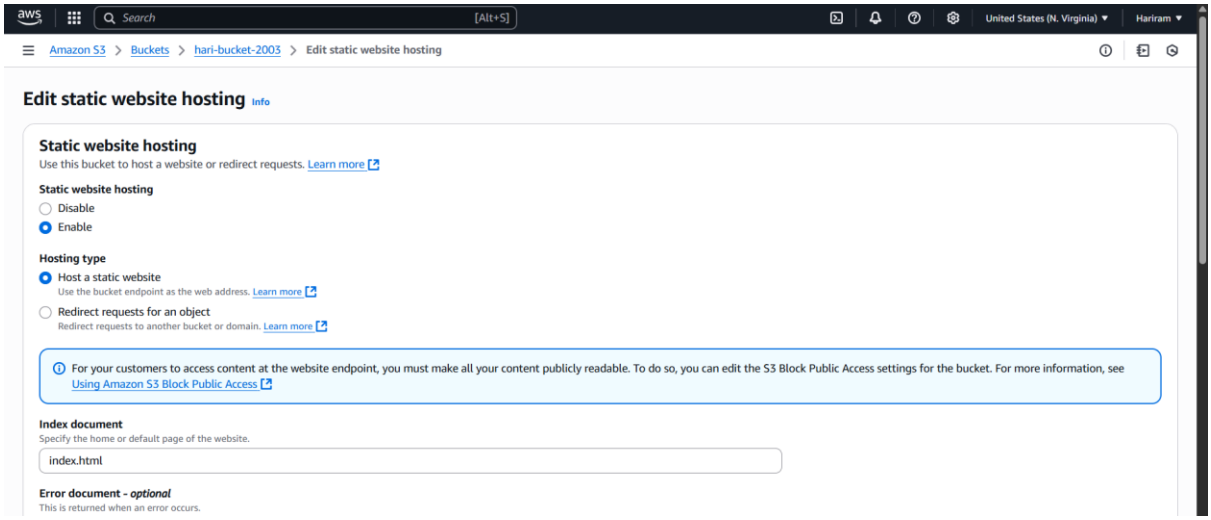
S3 bucket is created.



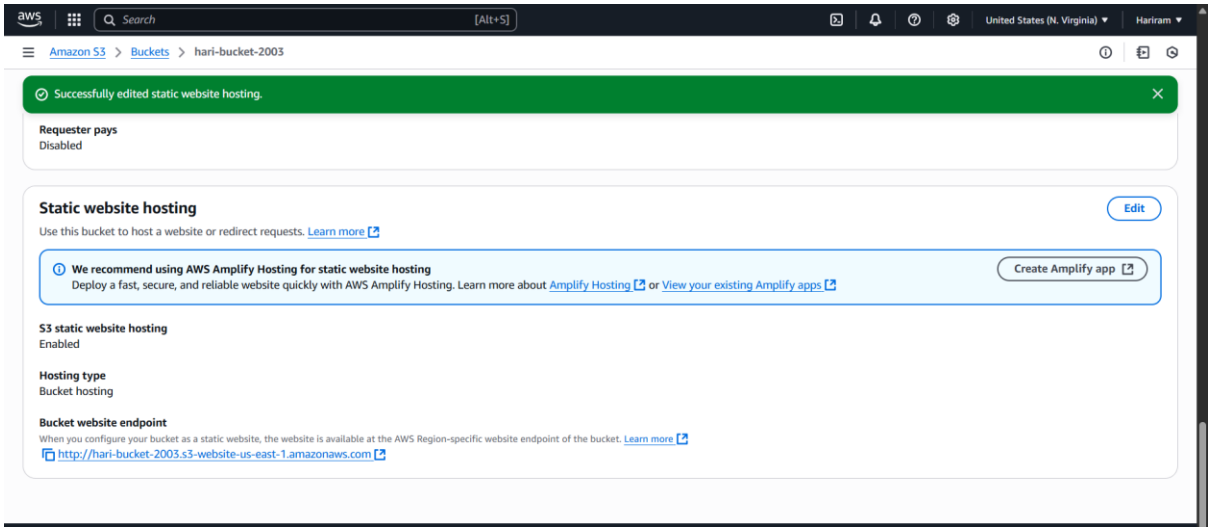
New bucket policy is added.



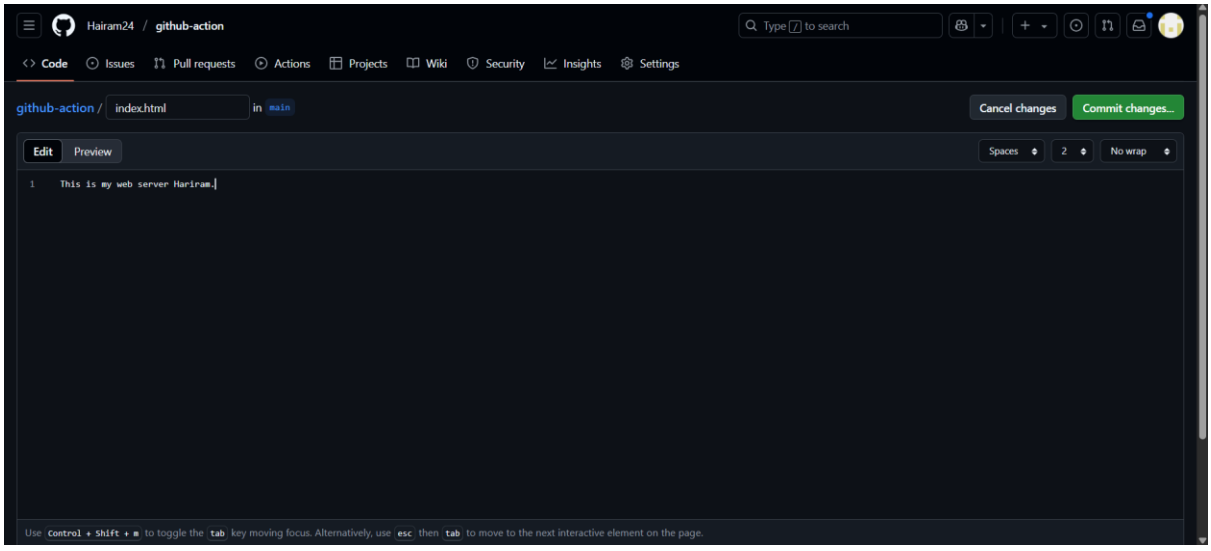
Bucket policy is added successfully.



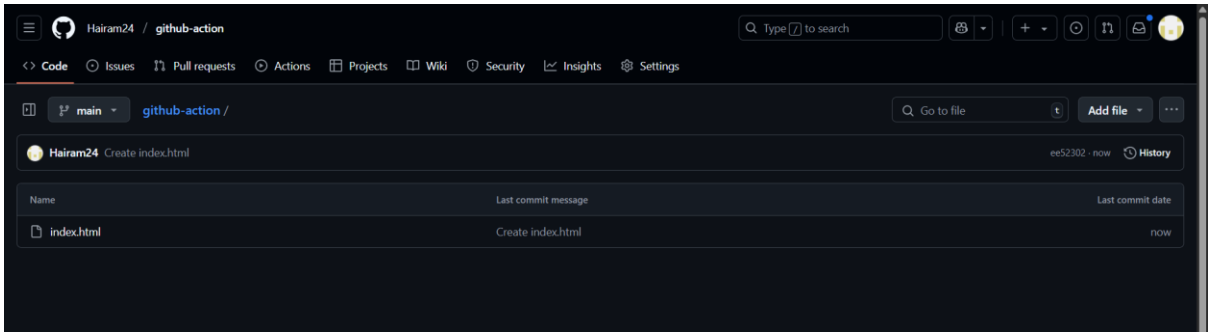
In s3 bucket , statis we hosting is enabled and index.html is added.



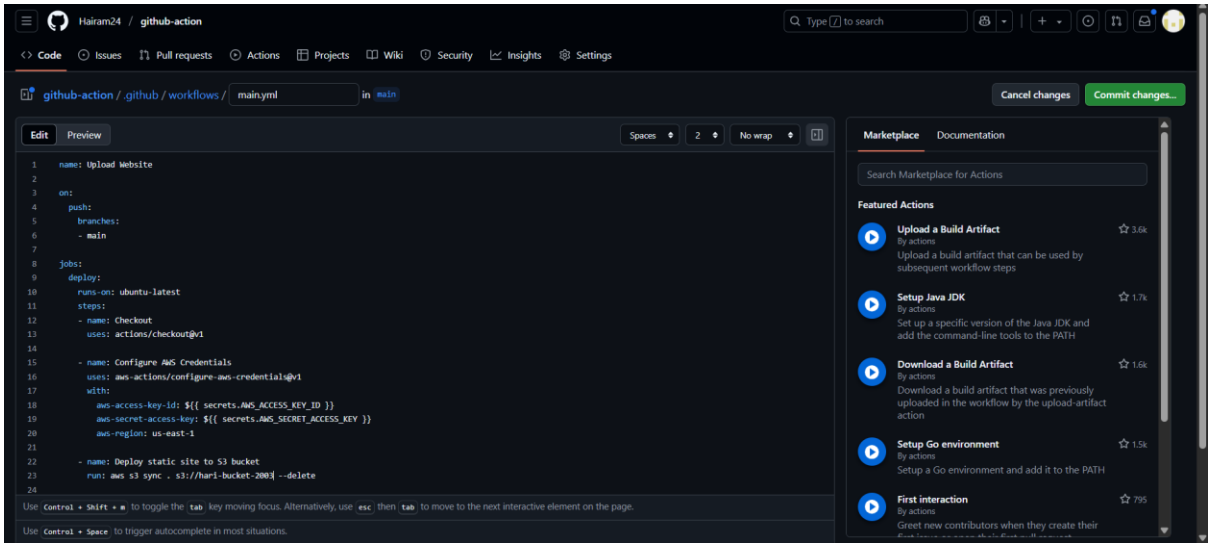
Static web hosting is enabled successfully.



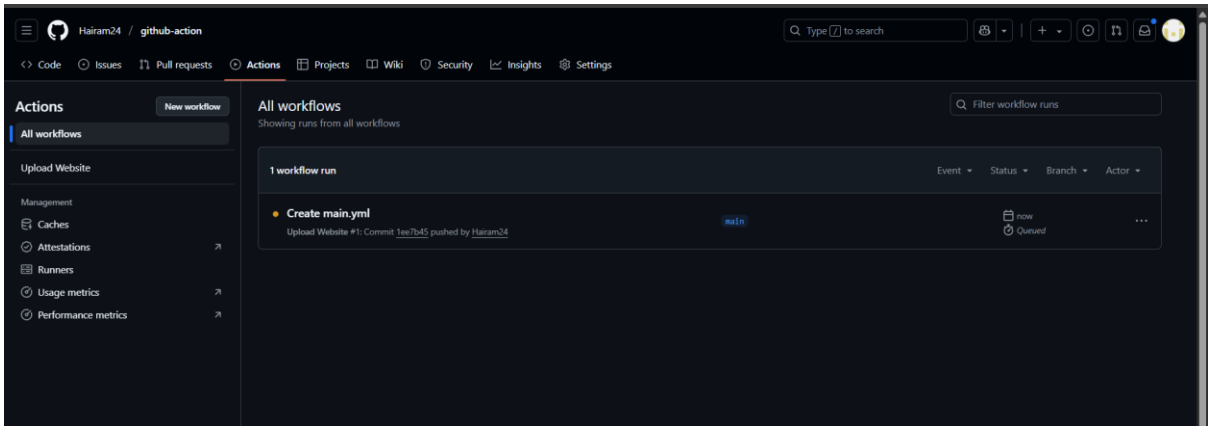
In the repo index.html file is created.



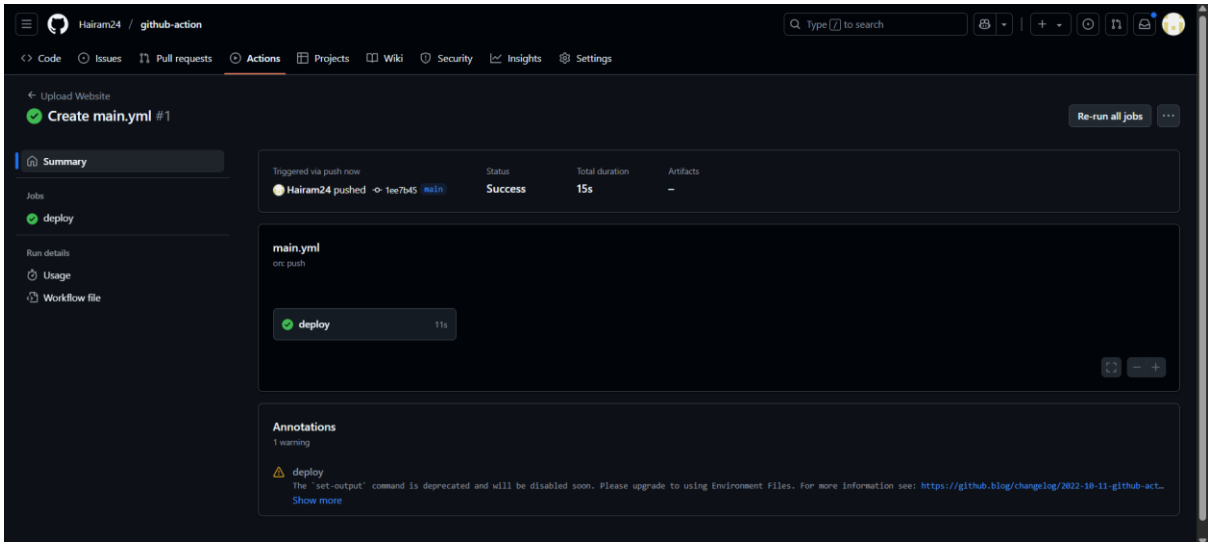
Index.html is added successfully.



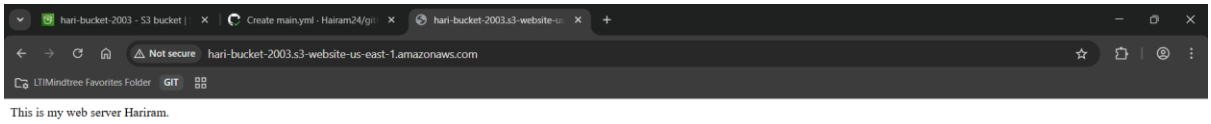
Main.yaml is added with my bucket name.



Yaml file is getting executed in the actions.



Statis website is deployed.

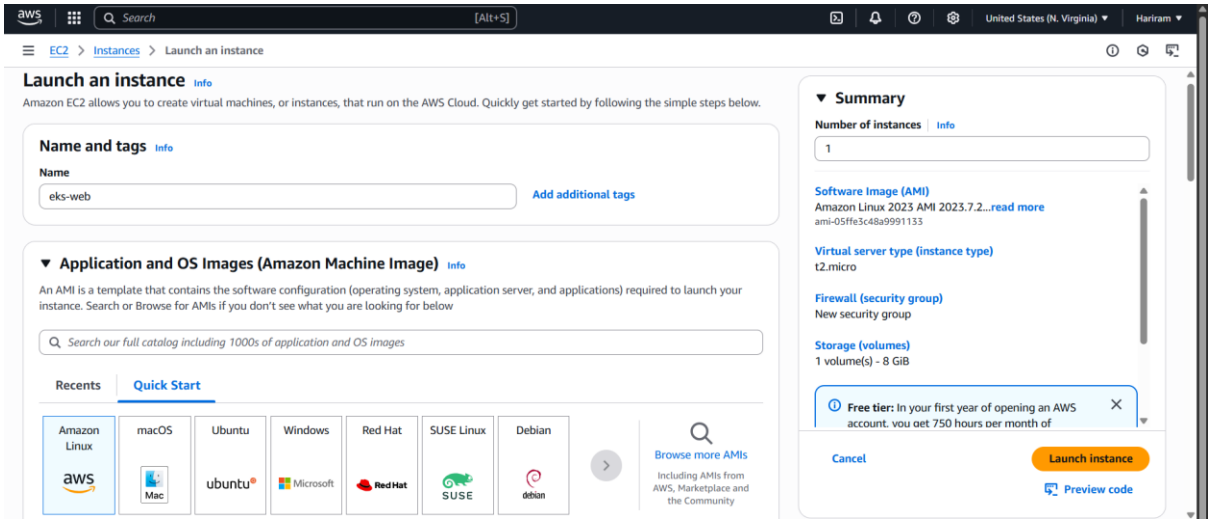


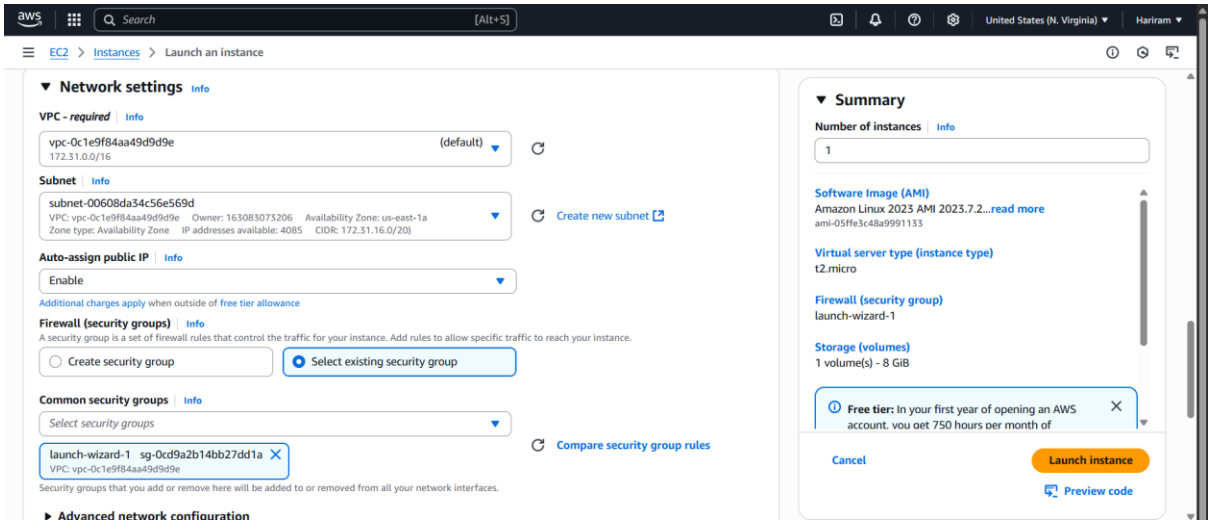
To view the website past the static web hosting link in the browser.

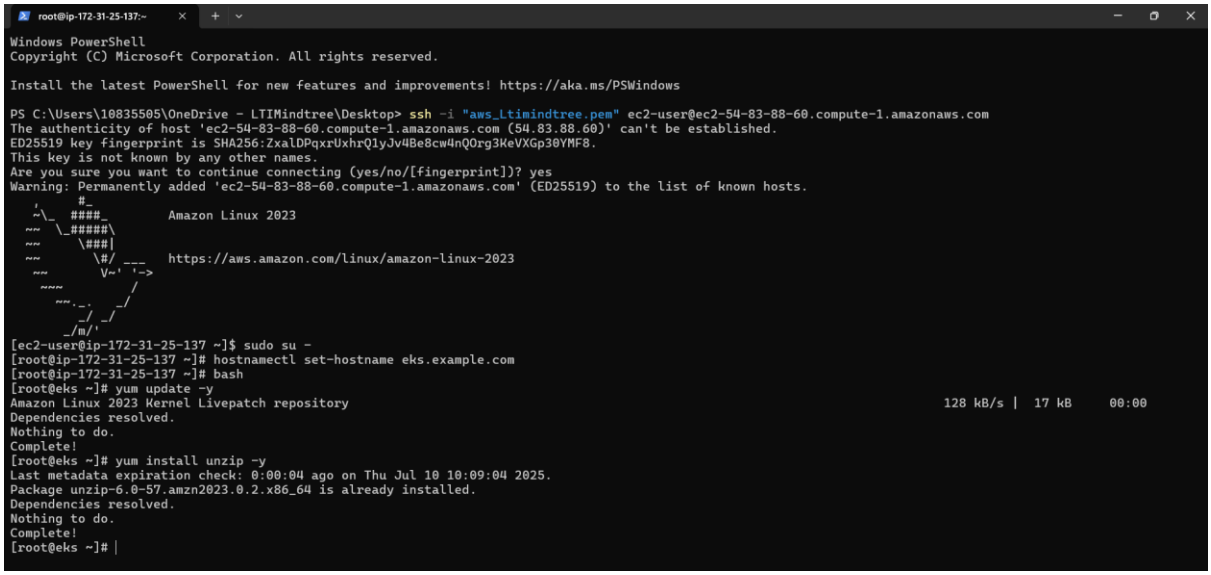
Static website is hosted successfully using s3 bucket and allowed publically.

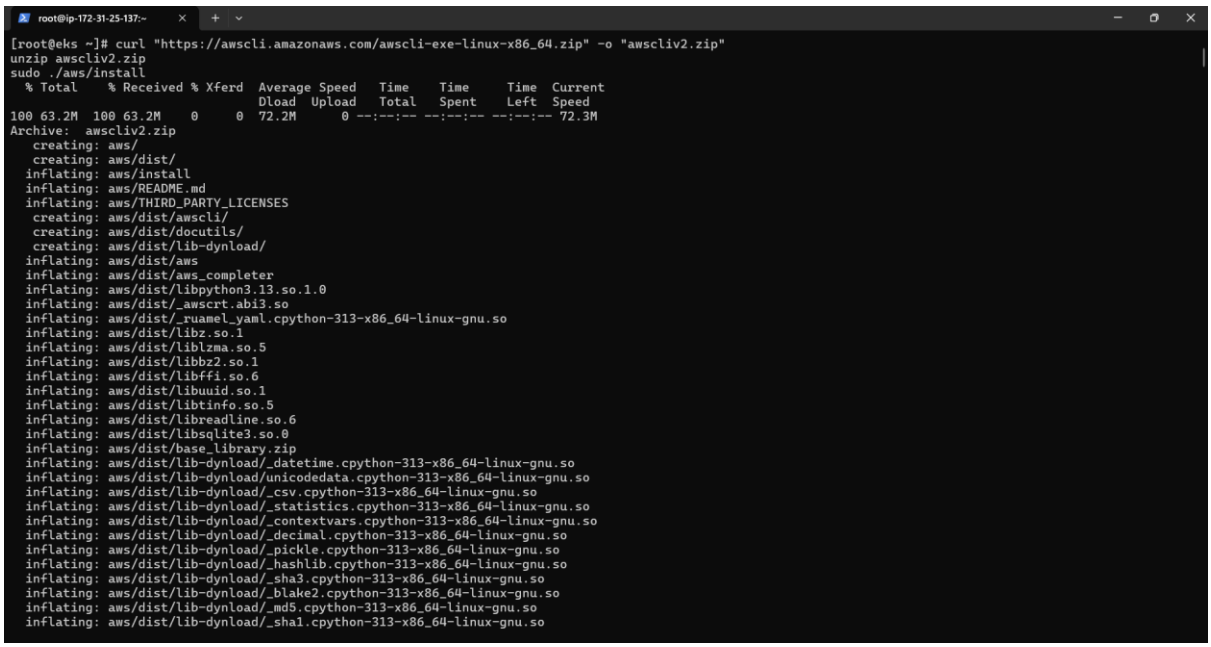
2) Medium

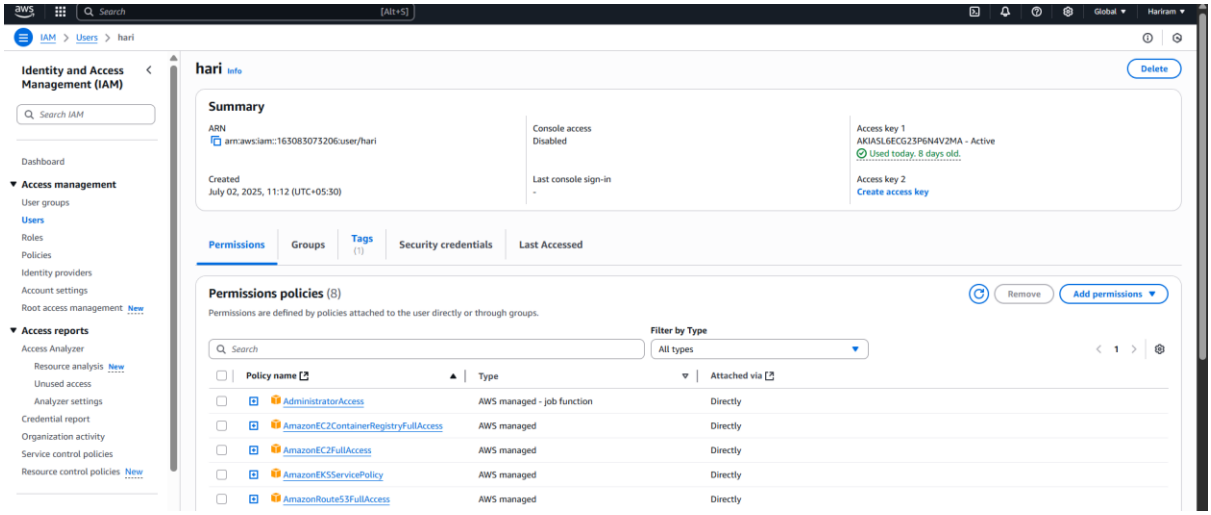
Deploy a web application in the kubernetes pod. And create a replica set. In any case load is going to increase on your replica set. increase the number of replica of the pods





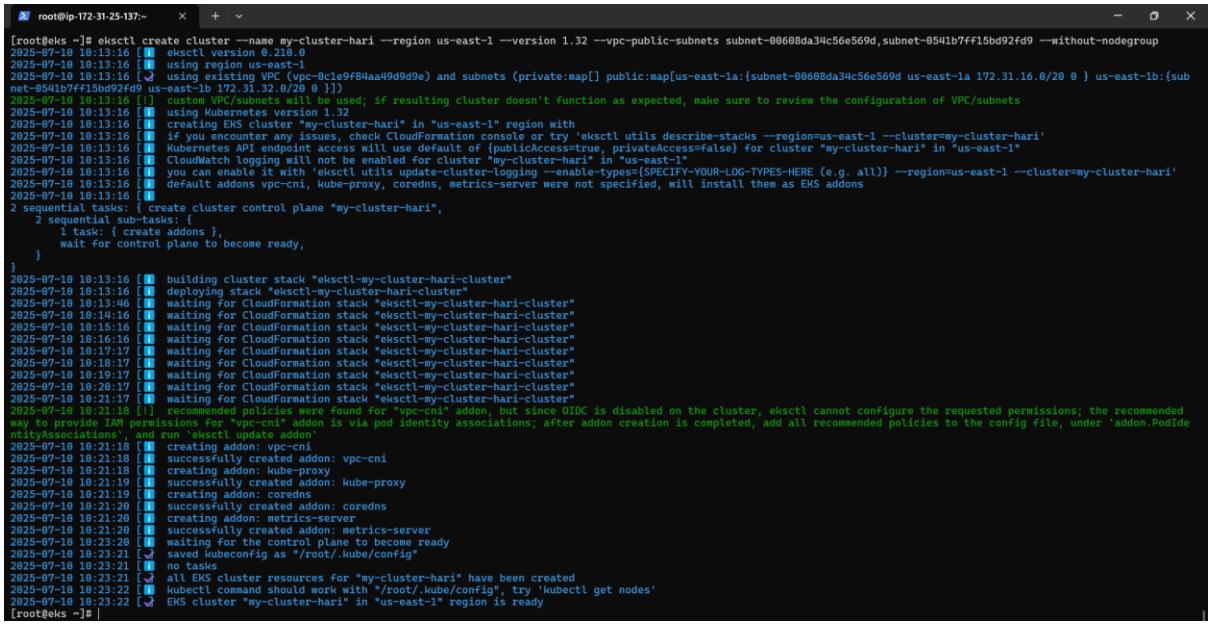

root@ip-172-31-25-137:~
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\10835505\OneDrive - LTIMindtree\Desktop> ssh -i "aws_Ltimindtree.pem" ec2-user@ec2-54-83-88-60.compute-1.amazonaws.com
The authenticity of host 'ec2-54-83-88-60.compute-1.amazonaws.com (54.83.88.60)' can't be established.
ED25519 key fingerprint is SHA256:ZxalDPqxrUxhrQ1yJv4Be8cw4nQOrg3KeVXGp30YMF8.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-83-88-60.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023
[ec2-user@ip-172-31-25-137 ~]$ sudo su -
[root@ip-172-31-25-137 ~]# hostnamectl set-hostname eks.example.com
[root@ip-172-31-25-137 ~]# bash
[root@eks ~]# yum update -y
Amazon Linux 2023 Kernel Livepatch repository 128 kB/s | 17 kB 00:00
Dependencies resolved.
Nothing to do.
Complete!
[root@eks ~]# yum install unzip -y
Last metadata expiration check: 0:00:04 ago on Thu Jul 10 10:09:04 2025.
Package unzip-6.0-57.amzn2023.0.2.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@eks ~]#


root@ip-172-31-25-137:~
[root@eks ~]# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 63.2M 100 63.2M 0 0 72.2M 0 --:--:-- --:--:-- --:--:-- 72.3M
Archive: awscliv2.zip
creating: aws/
creating: aws/dist/
inflating: aws/install
inflating: aws/README.md
inflating: aws/THIRD_PARTY_LICENSES
creating: aws/dist/awscli/
creating: aws/dist/docutils/
creating: aws/dist/lib-dynload/
inflating: aws/dist/aws
inflating: aws/dist/aws_completer
inflating: aws/dist/libpython3.13.so.1.0
inflating: aws/dist/_awscrt.abi3.so
inflating: aws/dist/_ruamel_yaml.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/libz.so.1
inflating: aws/dist/liblzma.so.5
inflating: aws/dist/libbz2.so.1
inflating: aws/dist/libffi.so.6
inflating: aws/dist/libuuid.so.1
inflating: aws/dist/libtinfo.so.5
inflating: aws/dist/libreadline.so.6
inflating: aws/dist/libsqlite3.so.0
inflating: aws/dist/base_library.zip
inflating: aws/dist/lib-dynload/_datetime.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/unicodedata.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_csv.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_statistics.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_contextvars.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_decimal.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_pickle.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_hashlib.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_sha3.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_blake2.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_md5.cpython-313-x86_64-linux-gnu.so
inflating: aws/dist/lib-dynload/_sha1.cpython-313-x86_64-linux-gnu.so

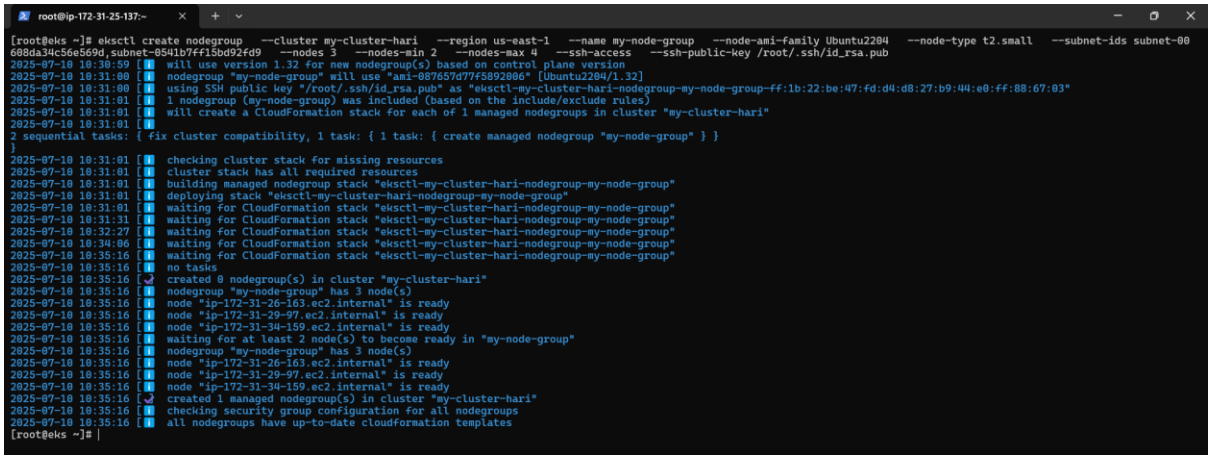



```
[root@eks ~]# aws configure
AWS Access Key ID [None]: AKIASL6ECG23P6N4V2MA
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: table
[root@eks ~]#
```

```
root@ip-172-31-25-137:~
[root@eks ~]# curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
[root@eks ~]# sudo mv /tmp/eksctl /usr/local/bin
[root@eks ~]# eksctl version
0.210.0
[root@eks ~]# curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.
txt)/bin/linux/amd64/kubectl
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 53.7M  100 53.7M    0     0  93.3M      0 --:--:-- --:--:-- --:--:-- 93.3M
[root@eks ~]# sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
[root@eks ~]# kubectl version --client
Client Version: v1.31.0
Kustomize Version: v5.4.2
[root@eks ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Eqa5Mcf58+Setm4e4WqYz8a7PgjttkMYPDmGyt8/y2s root@eks.example.com
The key's randomart image is:
+---[RSA 3072]----+
|                 |
|                 |
|   o .o          |
|  . B= o         |
|.. .=*= S .      |
|..  o=oo . .     |
| . o+ =o +       |
|  . .OE=*oo      |
|    .+%@XO.      |
+----[SHA256]-----+
[root@eks ~]#
```

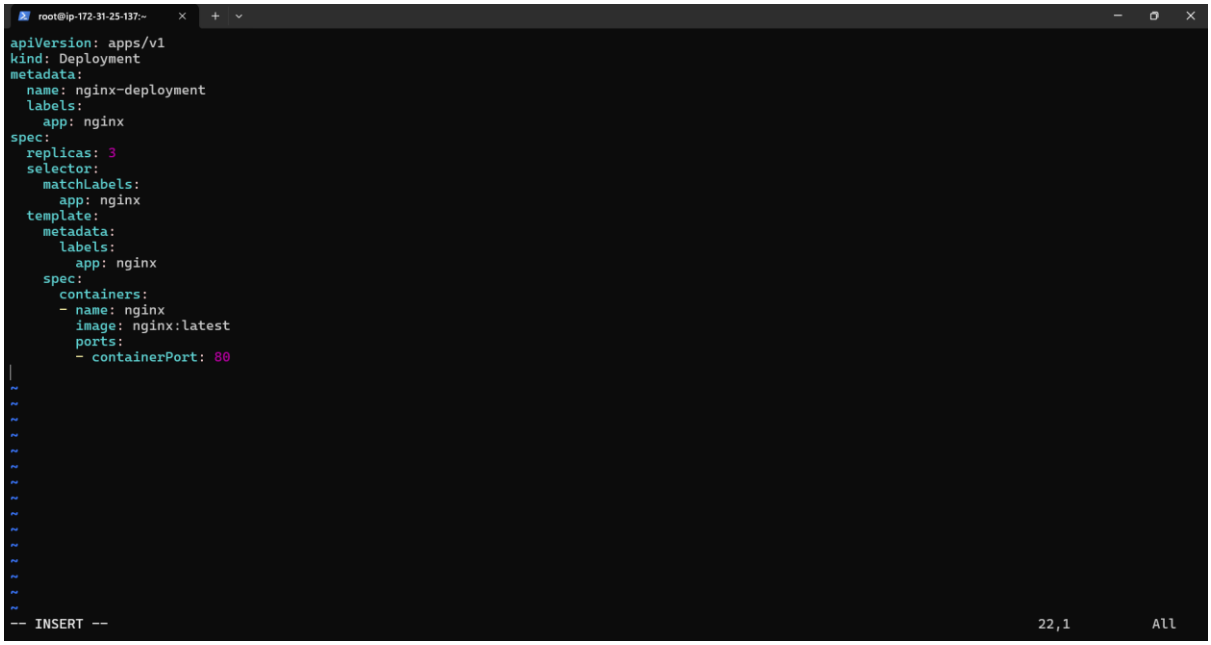


Cluster is created.



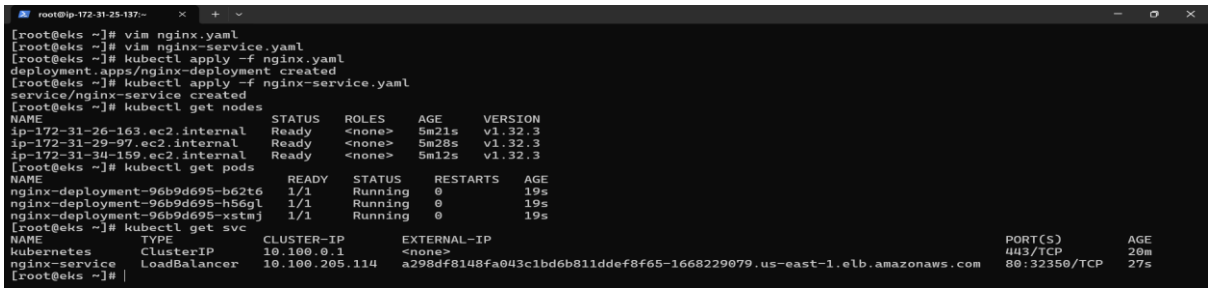
Node group is created.

Replica is automated based on load.

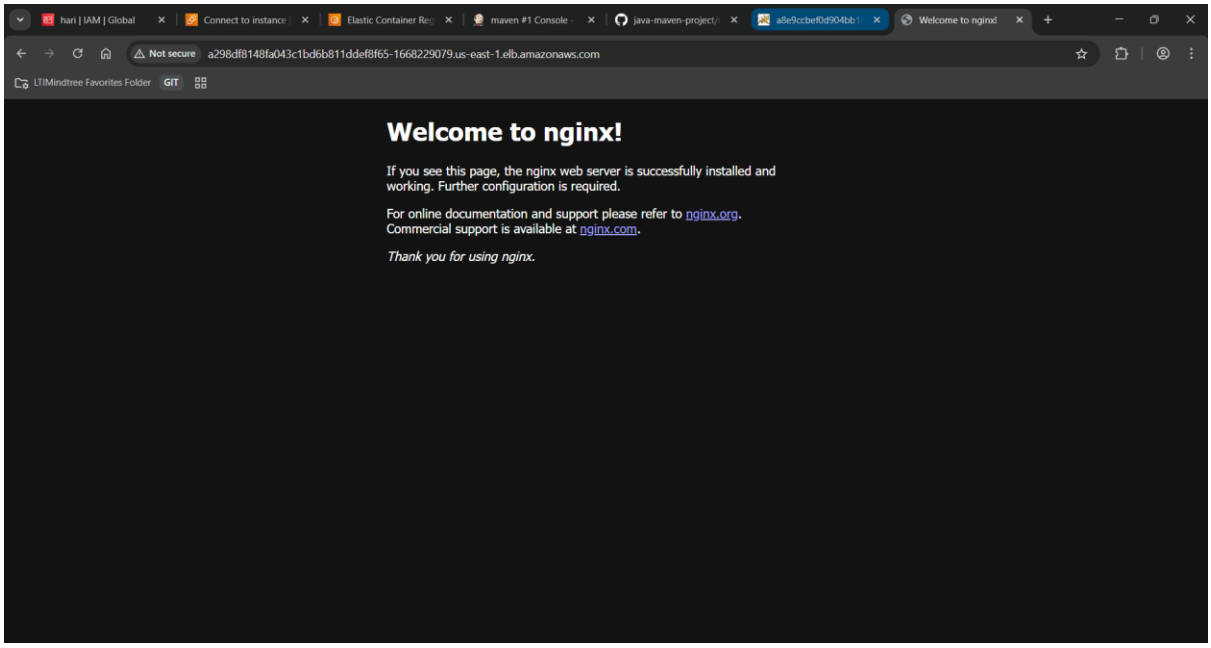


Nginx is deployed in yaml.

Service.yaml file is added.



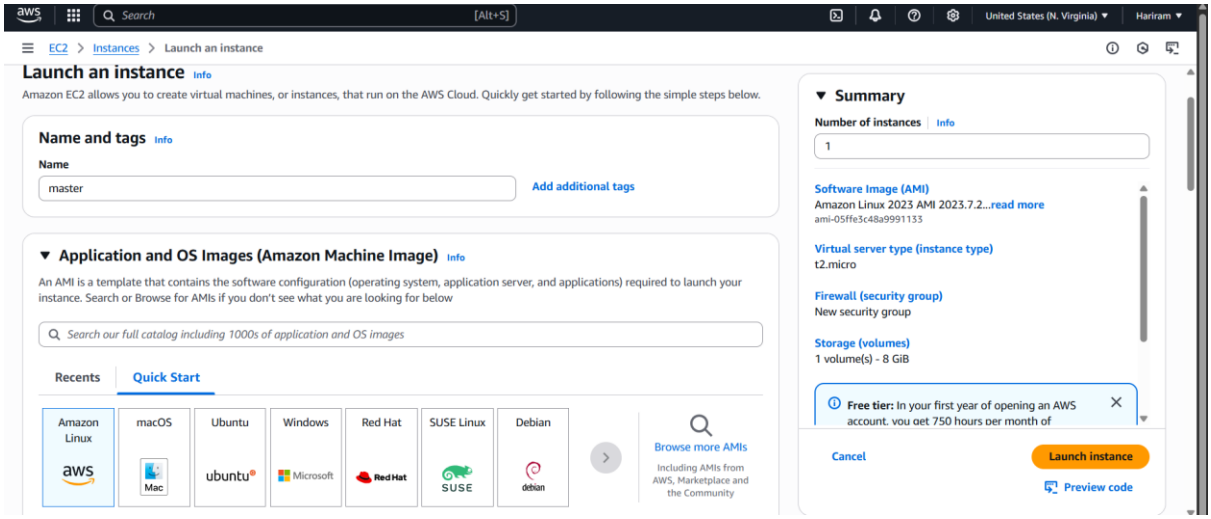
Pods are created successfully and external ip is taken.



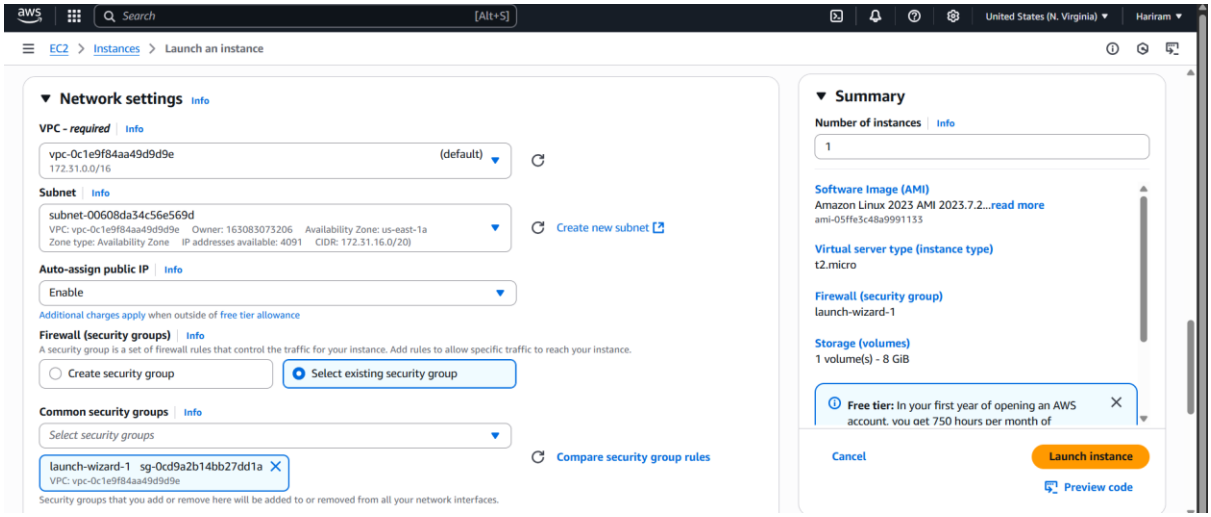
Nginx web app is deployed in the kubernetes pod and based on the load replica set will be dynamically added and deleted.

3) Easy-1

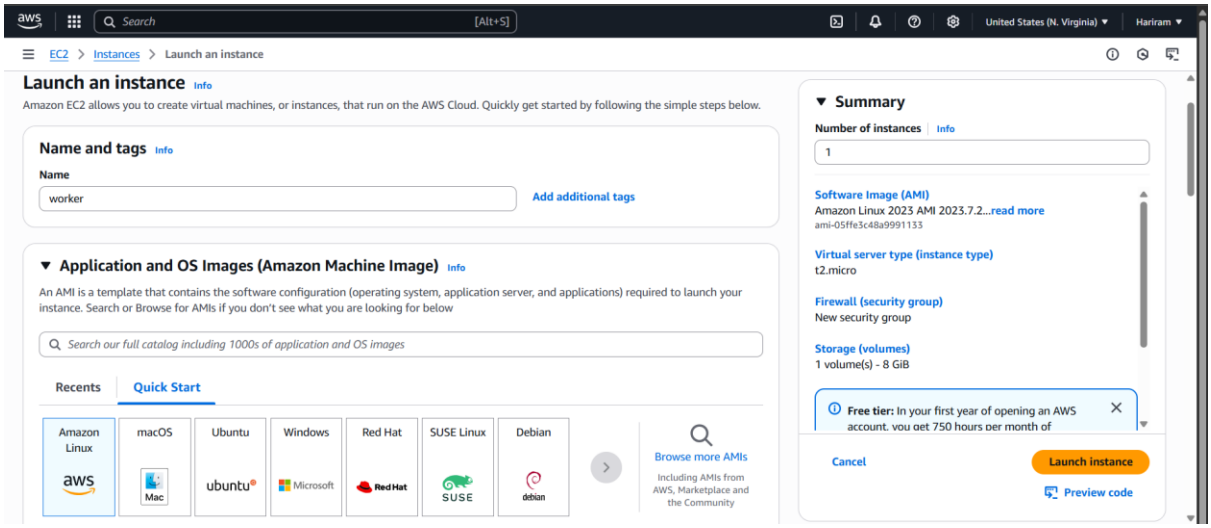
Using configuration management tool Ansible create multiple users and confirm on your manage host.



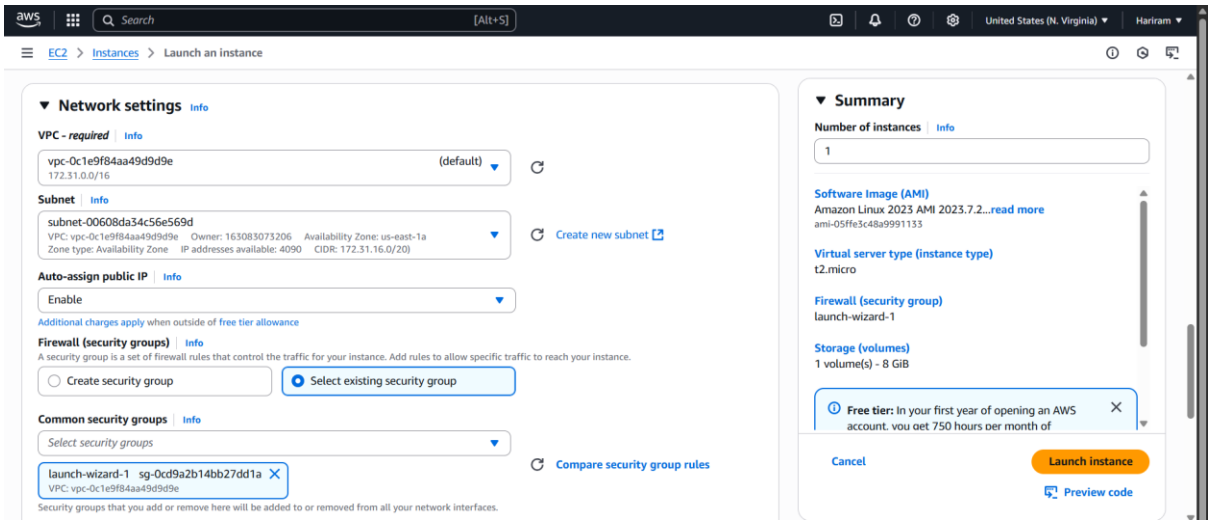
Master instance is created.



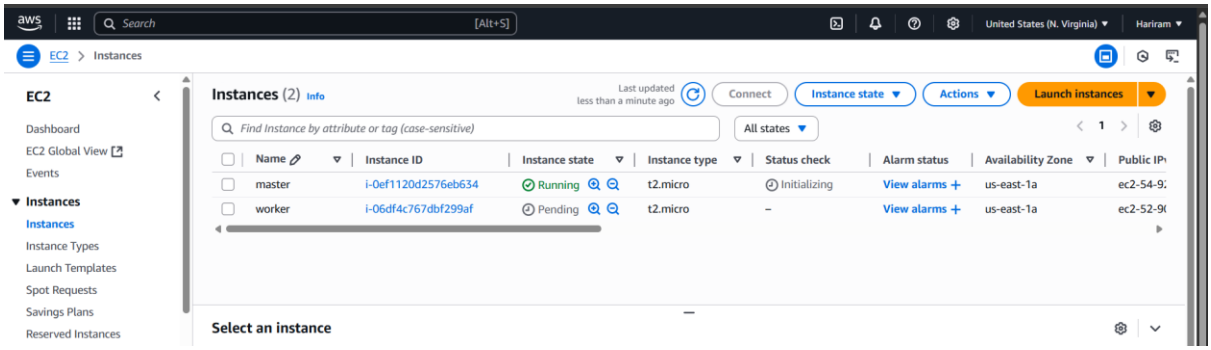
Existing key pair is selected and existing security group is selected.



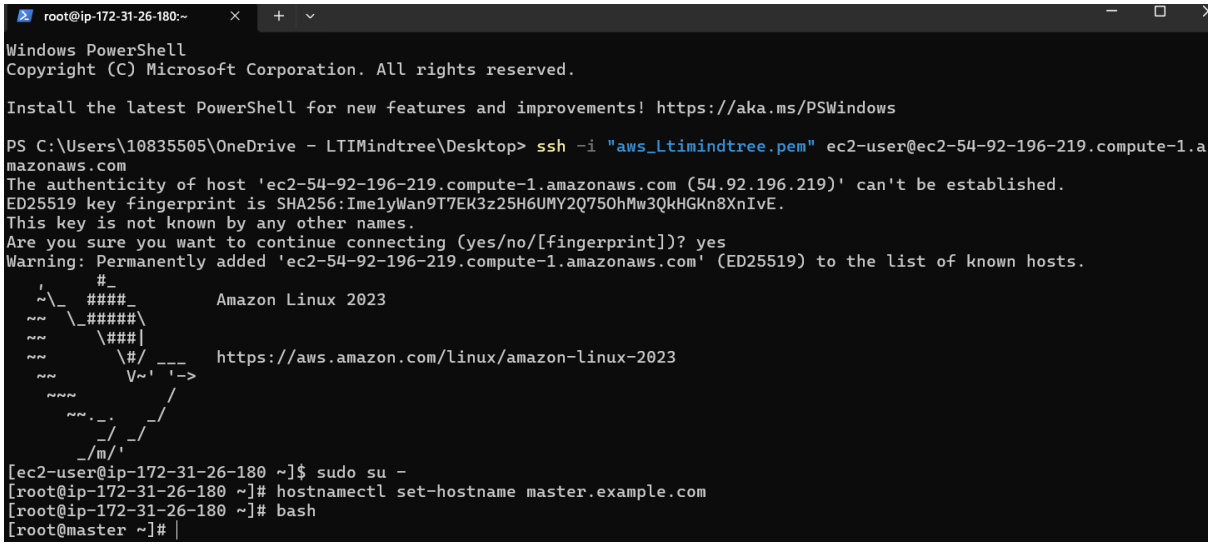
Worker node instance is created.



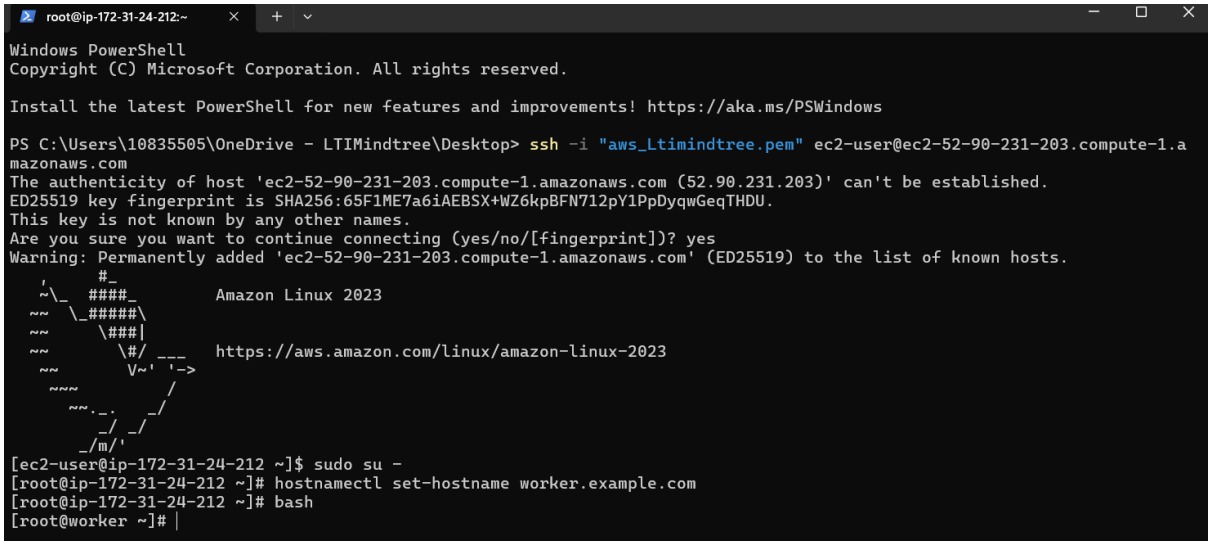
Existing key pair is selected and existing security group is selected.



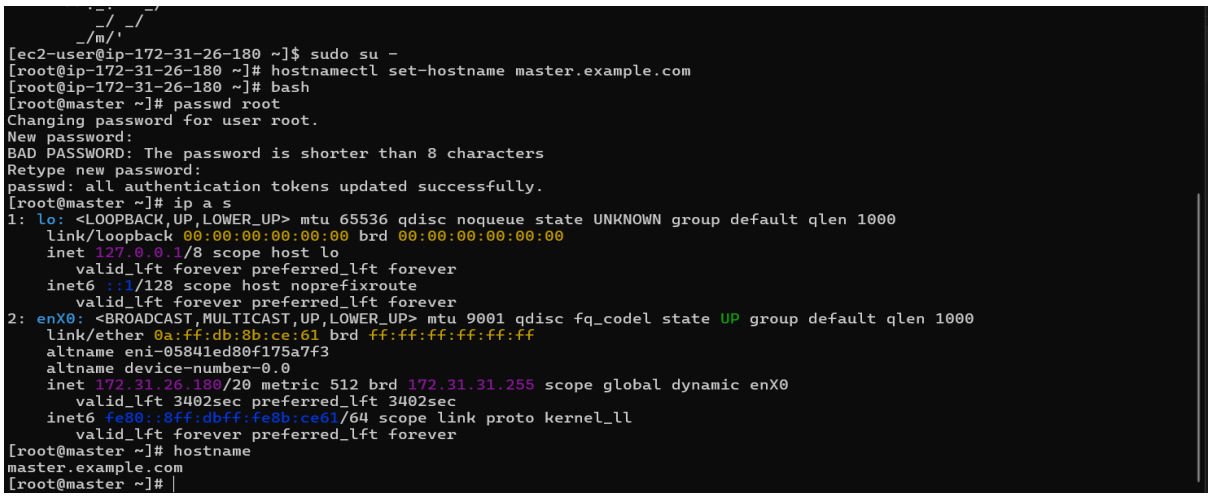
Instances are created successfully.



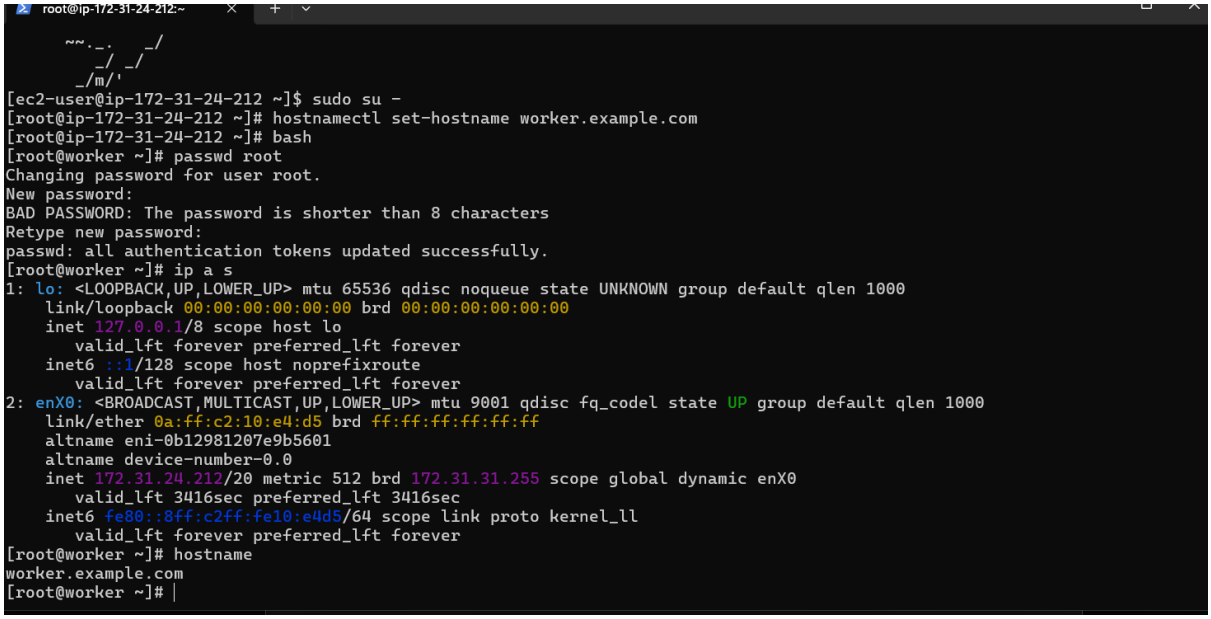
Master node is ready.



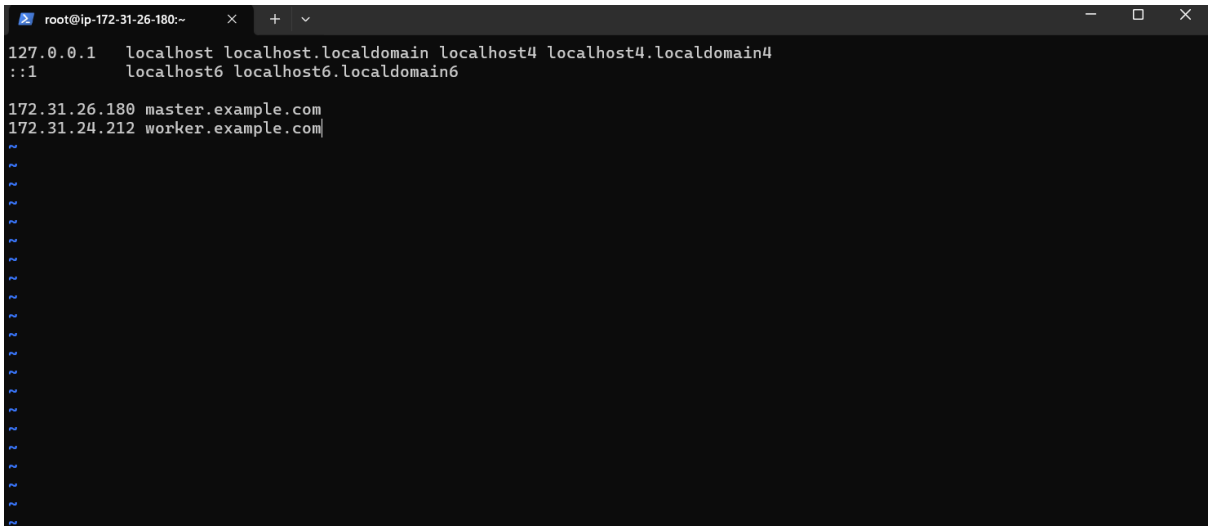
Worker node is ready.

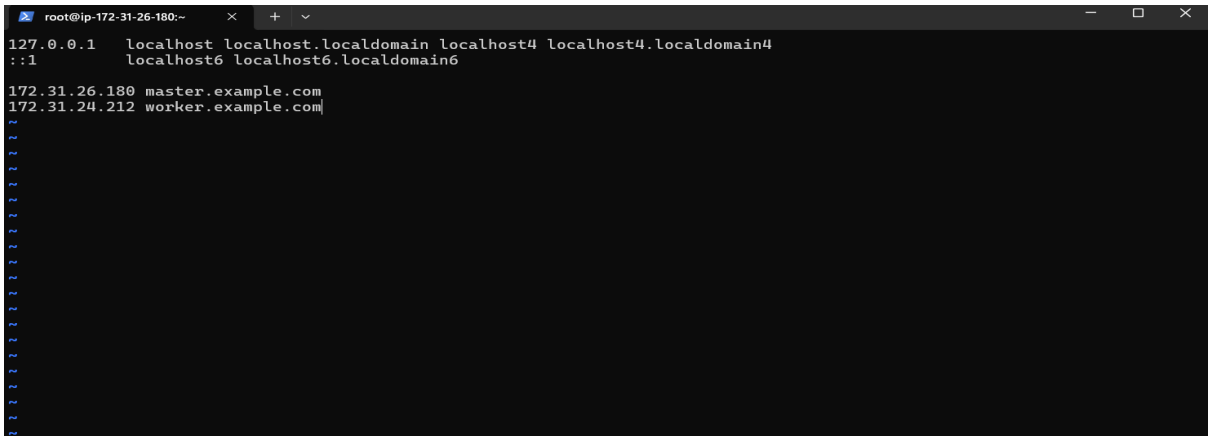


Ip and hostname of master is taken.

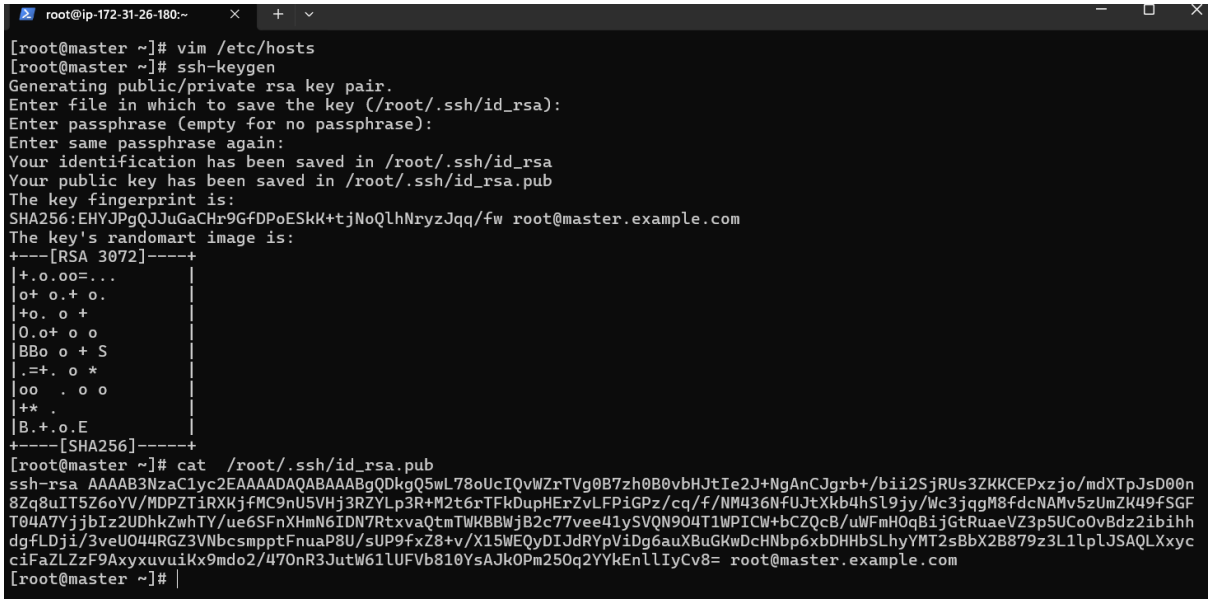


Ip and hostname of worker is taken.





Both machine ip and hostnames are shared.



Ssh-keygen is done and pub key is copied.

Pasted the pub key in worker node.

Pasted successfully.

```
root@ip-172-31-26-180:~
[root@master ~]# yum install ansible*
Amazon Linux 2023 Kernel Livepatch repository                                                    149 kB/s |  17 kB     00:00
Last metadata expiration check: 0:00:01 ago on Thu Jul 10 09:03:20 2025.
Dependencies resolved.
================================================================================================================================================
 Package                                  Architecture             Version                                  Repository                      Size
================================================================================================================================================
Installing:
 ansible                                  noarch                   8.3.0-1.amzn2023.0.1                     amazonlinux                     32 M
 ansible-core                             x86_64                   2.15.3-1.amzn2023.0.11                   amazonlinux                    2.5 M
 ansible-packaging                        noarch                   1-11.amzn2023.0.1                        amazonlinux                     14 k
 ansible-packaging-tests                  noarch                   1-11.amzn2023.0.1                        amazonlinux                    9.7 k
 ansible-srpm-macros                      noarch                   1-11.amzn2023.0.1                        amazonlinux                     22 k
Installing dependencies:
 ansible-test                             x86_64                   2.15.3-1.amzn2023.0.11                   amazonlinux                    704 k
 git-core                                 x86_64                   2.47.1-1.amzn2023.0.3                    amazonlinux                    4.5 M
 python3-apipkg                           noarch                   1.5-12.amzn2023.0.2                      amazonlinux                     19 k
 python3-execnet                          noarch                   1.7.1-5.amzn2023.0.2                     amazonlinux                    333 k
 python3-iniconfig                        noarch                   1.1.1-2.amzn2023.0.2                     amazonlinux                     17 k
 python3-packaging                        noarch                   21.3-2.amzn2023.0.2                      amazonlinux                     72 k
 python3-pluggy                           noarch                   0.13.1-3.amzn2023.0.2                    amazonlinux                     38 k
 python3-py                               noarch                   1.10.0-2.amzn2023.0.2                    amazonlinux                    428 k
 python3-pyparsing                        noarch                   2.4.7-6.amzn2023.0.2                     amazonlinux                    152 k
 python3-pytest                           noarch                   6.2.2-1.amzn2023.0.3                     amazonlinux                    1.5 M
 python3-pytest-forked                    noarch                   1.3.0-2.amzn2023.0.2                     amazonlinux                     17 k
 python3-pytest-mock                      noarch                   3.5.1-2.amzn2023.0.2                     amazonlinux                     32 k
 python3-pytest-xdist                     noarch                   2.2.0-2.amzn2023.0.2                     amazonlinux                     69 k
 python3-toml                             noarch                   0.10.2-2.amzn2023.0.2                    amazonlinux                     39 k
 sshpass                                  x86_64                   1.09-6.amzn2023.0.1                      amazonlinux                     28 k

Transaction Summary
================================================================================================================================================
Install  20 Packages
```

Ansible is installed in master node.

```
root@ip-172-31-26-180:~
  Verifying        : ansible-core-2.15.3-1.amzn2023.0.11.x86_64                                                                  2/20
  Verifying        : ansible-packaging-1-11.amzn2023.0.1.noarch                                                                  3/20
  Verifying        : ansible-packaging-tests-1-11.amzn2023.0.1.noarch                                                            4/20
  Verifying        : ansible-srpm-macros-1-11.amzn2023.0.1.noarch                                                                5/20
  Verifying        : ansible-test-2.15.3-1.amzn2023.0.11.x86_64                                                                  6/20
  Verifying        : git-core-2.47.1-1.amzn2023.0.3.x86_64                                                                       7/20
  Verifying        : python3-apipkg-1.5-12.amzn2023.0.2.noarch                                                                   8/20
  Verifying        : python3-execnet-1.7.1-5.amzn2023.0.2.noarch                                                                 9/20
  Verifying        : python3-iniconfig-1.1.1-2.amzn2023.0.2.noarch                                                              10/20
  Verifying        : python3-packaging-21.3-2.amzn2023.0.2.noarch                                                               11/20
  Verifying        : python3-pluggy-0.13.1-3.amzn2023.0.2.noarch                                                                12/20
  Verifying        : python3-py-1.10.0-2.amzn2023.0.2.noarch                                                                    13/20
  Verifying        : python3-pyparsing-2.4.7-6.amzn2023.0.2.noarch                                                              14/20
  Verifying        : python3-pytest-6.2.2-1.amzn2023.0.3.noarch                                                                 15/20
  Verifying        : python3-pytest-forked-1.3.0-2.amzn2023.0.2.noarch                                                          16/20
  Verifying        : python3-pytest-mock-3.5.1-2.amzn2023.0.2.noarch                                                            17/20
  Verifying        : python3-pytest-xdist-2.2.0-2.amzn2023.0.2.noarch                                                           18/20
  Verifying        : python3-toml-0.10.2-2.amzn2023.0.2.noarch                                                                  19/20
  Verifying        : sshpass-1.09-6.amzn2023.0.1.x86_64                                                                         20/20

Installed:
  ansible-8.3.0-1.amzn2023.0.1.noarch                ansible-core-2.15.3-1.amzn2023.0.11.x86_64       ansible-packaging-1-11.amzn2023.0.1.noarch
  ansible-packaging-tests-1-11.amzn2023.0.1.noarch   ansible-srpm-macros-1-11.amzn2023.0.1.noarch     ansible-test-2.15.3-1.amzn2023.0.11.x86_64
  git-core-2.47.1-1.amzn2023.0.3.x86_64              python3-apipkg-1.5-12.amzn2023.0.2.noarch        python3-execnet-1.7.1-5.amzn2023.0.2.noarch
  python3-iniconfig-1.1.1-2.amzn2023.0.2.noarch      python3-packaging-21.3-2.amzn2023.0.2.noarch     python3-pluggy-0.13.1-3.amzn2023.0.2.noarch
  python3-py-1.10.0-2.amzn2023.0.2.noarch            python3-pyparsing-2.4.7-6.amzn2023.0.2.noarch    python3-pytest-6.2.2-1.amzn2023.0.3.noarch
  python3-pytest-forked-1.3.0-2.amzn2023.0.2.noarch  python3-pytest-mock-3.5.1-2.amzn2023.0.2.noarch  python3-pytest-xdist-2.2.0-2.amzn2023.0.2.noarch
  python3-toml-0.10.2-2.amzn2023.0.2.noarch          sshpass-1.09-6.amzn2023.0.1.x86_64

Complete!
[root@master ~]# ansible --version
ansible [core 2.15.3]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.9/site-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.9.23 (main, Jun 11 2025, 00:00:00) [GCC 11.5.0 20240719 (Red Hat 11.5.0-5)] (/usr/bin/python3.9)
  jinja version = 3.1.4
  libyaml = True
[root@master ~]#
```

Ansible is available in master node.

```
[root@master ~]# cd /etc/ansible
[root@master ansible]# ll
total 0
drwxr-xr-x. 2 root root 6 Mar 24 18:48 roles
[root@master ansible]# vim ansible.cfg
[root@master ansible]# ll
total 8
-rw-r--r--. 1 root root 6119 Jul 10 09:05 ansible.cfg
drwxr-xr-x. 2 root root    6 Mar 24 18:48 roles
[root@master ansible]#
```

Ansible.cfg file is added.

```
root@ip-172-31-26-180:/etc/a
---
- name: create multiple users
  hosts: all
  tasks:
    - name: create multiple users
      user:
        name: "{{ item }}"
        state: present
      loop:
        - hariram
        - drcula
        - ironman
```

Playbook to add multiple use is added . Users hariram,Dracula,ironman will be added.

```
root@ip-172-31-26-180:/etc/a
[root@master ansible]# vim user.yaml
[root@master ansible]# ansible-playbook user.yaml

PLAY [create multiple users] ***************************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************************
The authenticity of host 'worker.example.com (172.31.24.212)' can't be established.
ED25519 key fingerprint is SHA256:65F1ME7a6iAEBSX+WZ6kpBFN712pY1PpDyqwGeqTHDU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
[WARNING]: Platform linux on host worker.example.com is using the discovered Python interpreter at /usr/bin/python3.9, but future installation of another
Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-core/2.15/reference_appendices/interpreter_discovery.html
for more information.
ok: [worker.example.com]

TASK [create multiple users] ***************************************************************************************************************************************
changed: [worker.example.com] => (item=hariram)
changed: [worker.example.com] => (item=drcula)
changed: [worker.example.com] => (item=ironman)

PLAY RECAP *********************************************************************************************************************************************************
worker.example.com         : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[root@master ansible]#
```

Playbook is applied.

```
root@ip-172-31-24-212:~
[root@worker ~]# vim /etc/passwd
[root@worker ~]#
```

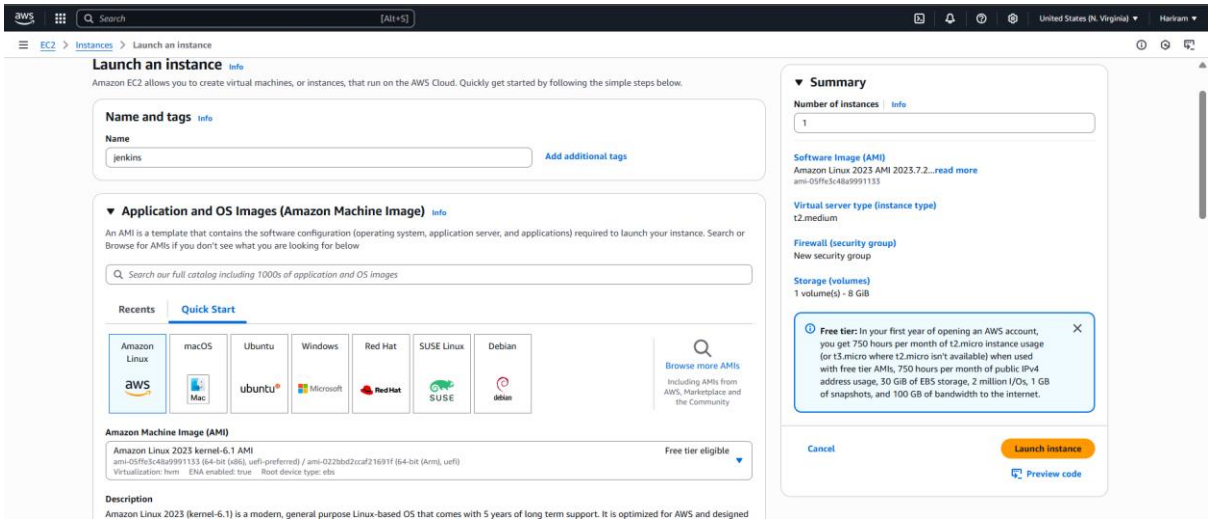
In worker node, using vim/etc/passwd , user added can be confirmed.

```
root@ip-172-31-24-212:~
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:65534:65534:Kernel Overflow User:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/:/usr/sbin/nologin
systemd-oom:x:999:999:systemd Userspace OOM Killer:/:/usr/sbin/nologin
systemd-resolve:x:193:193:systemd Resolver:/:/usr/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/usr/share/empty.sshd:/sbin/nologin
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
libstoragemgmt:x:997:997:daemon account for libstoragemgmt:/:/usr/sbin/nologin
systemd-coredump:x:996:996:systemd Core Dumper:/:/usr/sbin/nologin
systemd-timesync:x:995:995:systemd Time Synchronization:/:/usr/sbin/nologin
chrony:x:994:994:chrony system user:/var/lib/chrony:/sbin/nologin
ec2-instance-connect:x:993:993::/home/ec2-instance-connect:/sbin/nologin
stapunpriv:x:159:159:systemtap unprivileged user:/var/lib/stapunpriv:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
tcpdump:x:72:72::/:/sbin/nologin
ec2-user:x:1000:1000:EC2 Default User:/home/ec2-user:/bin/bash
hariram:x:1001:1001::/home/hariram:/bin/bash
drcula:x:1002:1002::/home/drcula:/bin/bash
ironman:x:1003:1003::/home/ironman:/bin/bash
"/etc/passwd" 31L, 1652B                                                                                                              1,1           All
```

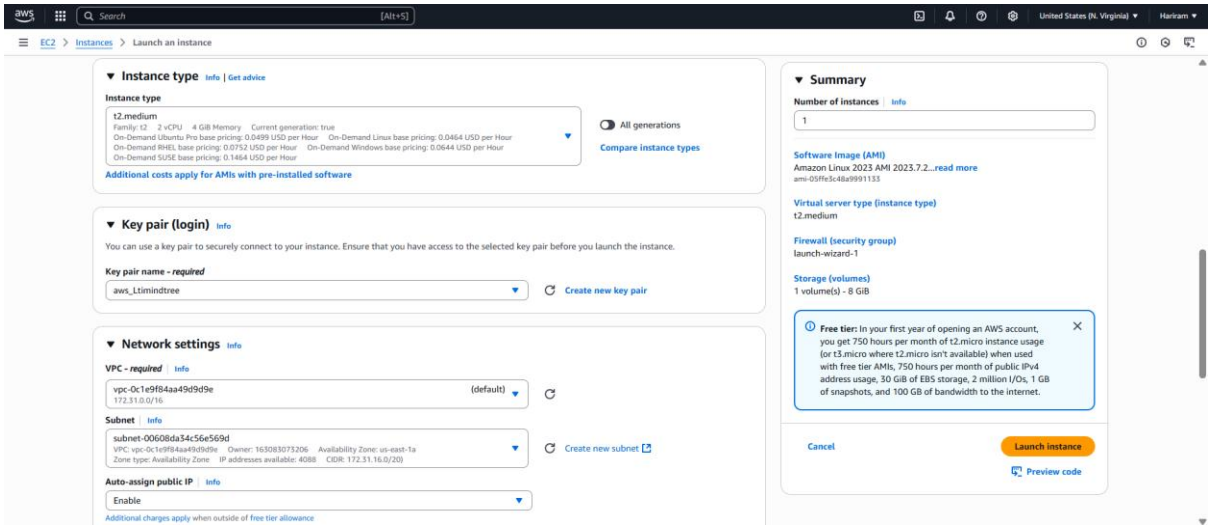
Multiple users added successfully using loop concept.

4)Medium-1

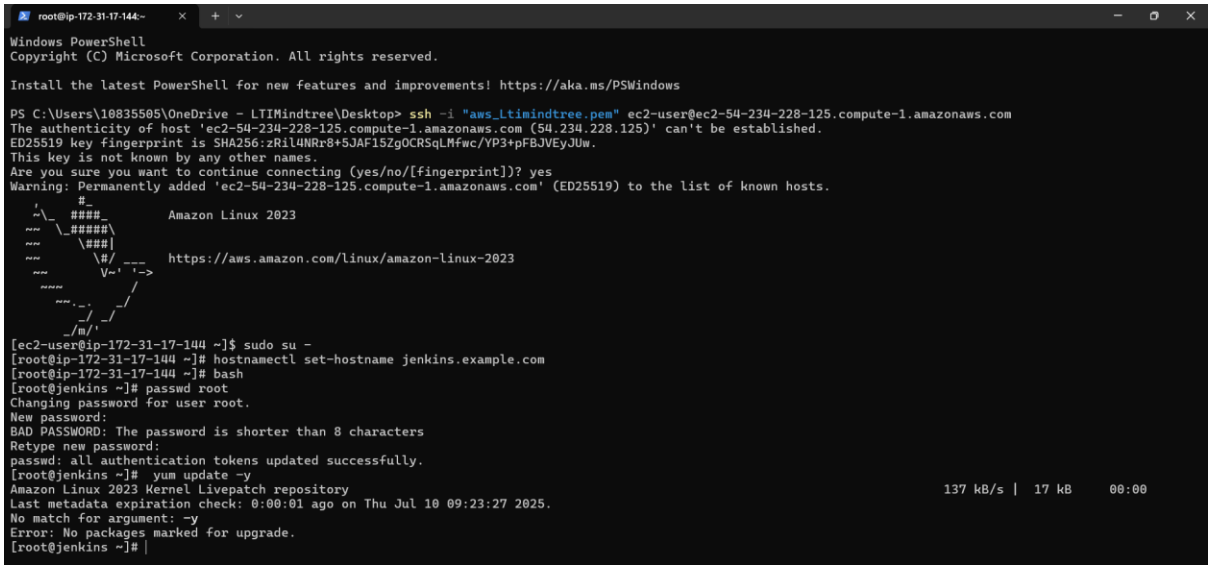
Deploy in a Docker instance and create a Docker image, Store the Docker image in ECR, achieve this image on EkS cluster using the ECR image. Build a sample java web app using maven



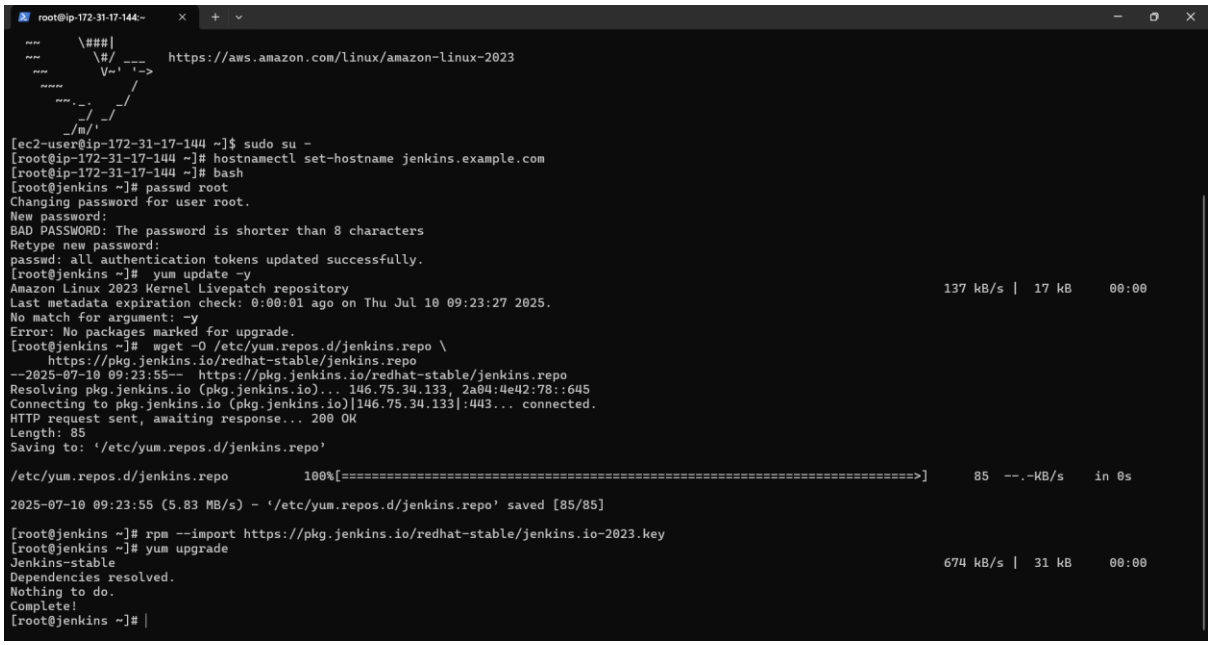
Jenkins instance is created.



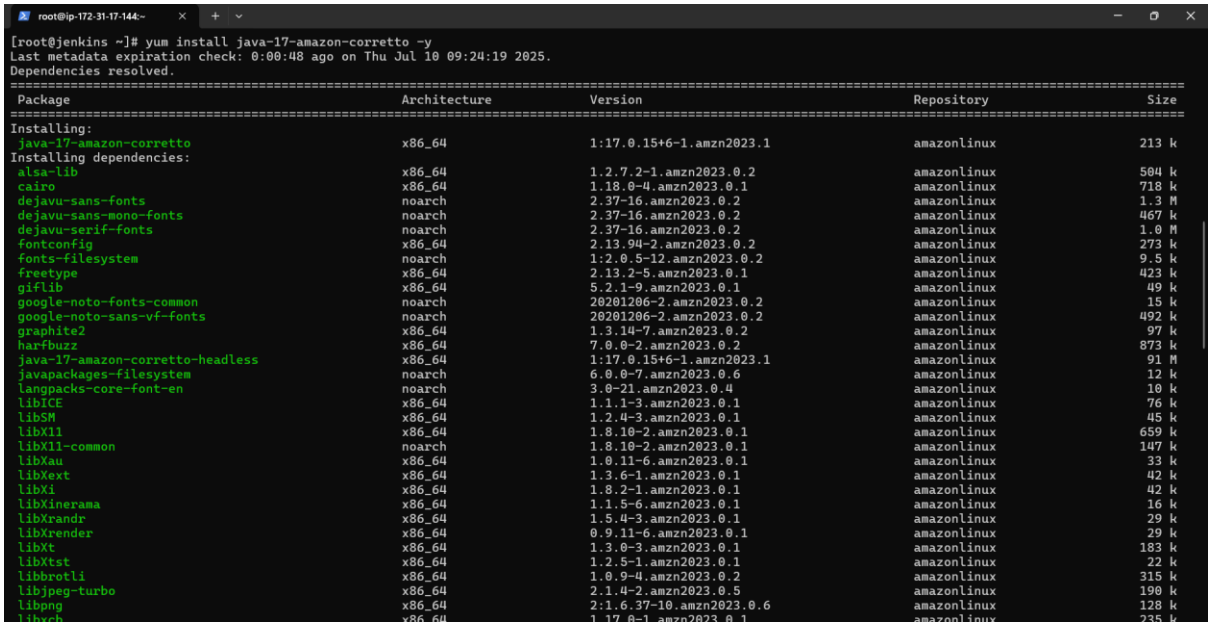
Existing key pair is selected . Existing security group is selected.



Jenkins is running successfully and necessary dependencies are installing.

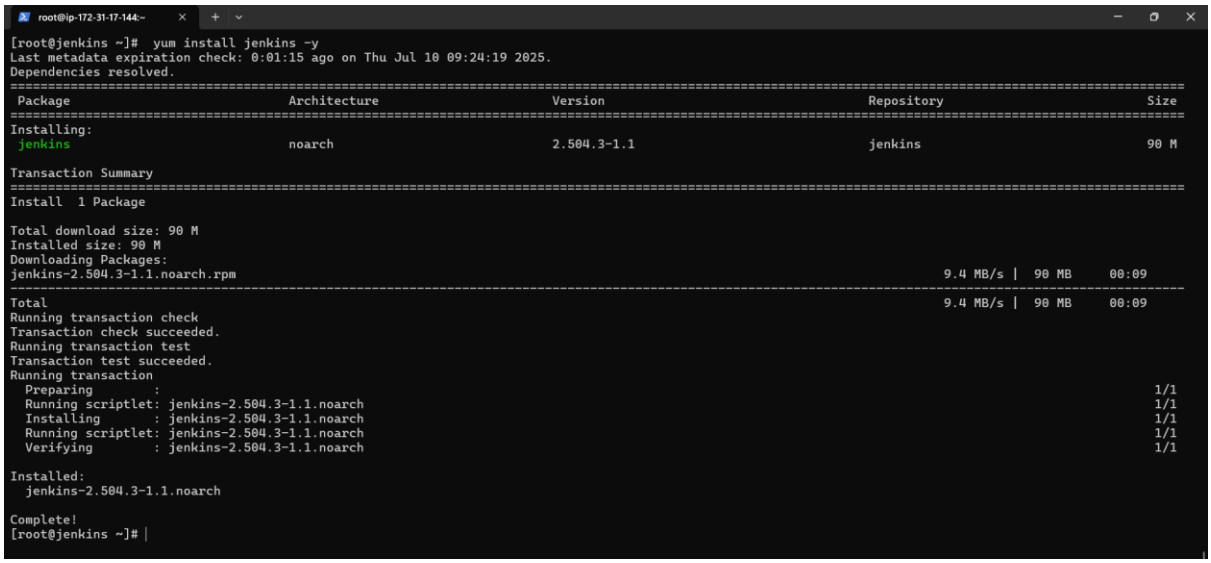


Yum upgrade is done.



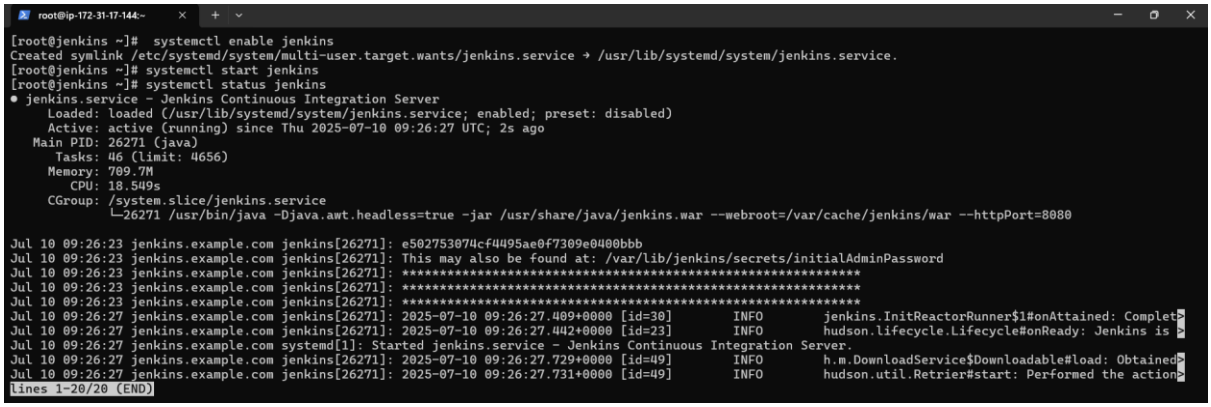
```
root@ip-172-31-17-144:~
[root@jenkins ~]# yum install java-17-amazon-corretto -y
Last metadata expiration check: 0:00:48 ago on Thu Jul 10 09:24:19 2025.
Dependencies resolved.
================================================================================
 Package                 Architecture  Version                    Repository  Size
================================================================================
Installing:
 java-17-amazon-corretto x86_64        1:17.0.15+6-1.amzn2023.1   amazonlinux 213 k
Installing dependencies:
 alsa-lib                x86_64        1.2.7.2-1.amzn2023.0.2     amazonlinux 504 k
 cairo                   x86_64        1.18.0-4.amzn2023.0.1      amazonlinux 718 k
 dejavu-sans-fonts       noarch        2.37-16.amzn2023.0.2       amazonlinux 1.3 M
 dejavu-sans-mono-fonts  noarch        2.37-16.amzn2023.0.2       amazonlinux 467 k
 dejavu-serif-fonts      noarch        2.37-16.amzn2023.0.2       amazonlinux 1.0 M
 fontconfig              x86_64        2.13.94-2.amzn2023.0.2     amazonlinux 273 k
 fonts-filesystem        noarch        1:2.0.5-12.amzn2023.0.2    amazonlinux 9.5 k
 freetype                x86_64        2.13.2-5.amzn2023.0.1      amazonlinux 423 k
 giflib                  x86_64        5.2.1-9.amzn2023.0.1       amazonlinux  49 k
 google-noto-fonts-common noarch       20201206-2.amzn2023.0.2    amazonlinux  15 k
 google-noto-sans-vf-fonts noarch      20201206-2.amzn2023.0.2    amazonlinux 492 k
 graphite2               x86_64        1.3.14-7.amzn2023.0.2      amazonlinux  97 k
 harfbuzz                x86_64        7.0.0-2.amzn2023.0.2       amazonlinux 873 k
 java-17-amazon-corretto-headless x86_64 1:17.0.15+6-1.amzn2023.1   amazonlinux  91 M
 javapackages-filesystem noarch        6.0.0-7.amzn2023.0.6       amazonlinux  12 k
 langpacks-core-font-en  noarch        3.0-21.amzn2023.0.4        amazonlinux  10 k
 libICE                  x86_64        1.1.1-3.amzn2023.0.1       amazonlinux  76 k
 libSM                   x86_64        1.2.4-3.amzn2023.0.1       amazonlinux  45 k
 libX11                  x86_64        1.8.10-2.amzn2023.0.1      amazonlinux 659 k
 libX11-common           noarch        1.8.10-2.amzn2023.0.1      amazonlinux 147 k
 libXau                  x86_64        1.0.11-6.amzn2023.0.1      amazonlinux  33 k
 libXext                 x86_64        1.3.6-1.amzn2023.0.1       amazonlinux  42 k
 libXi                   x86_64        1.8.2-1.amzn2023.0.1       amazonlinux  42 k
 libXinerama             x86_64        1.1.5-6.amzn2023.0.1       amazonlinux  16 k
 libXrandr               x86_64        1.5.4-3.amzn2023.0.1       amazonlinux  29 k
 libXrender              x86_64        0.9.11-6.amzn2023.0.1      amazonlinux  29 k
 libXt                   x86_64        1.3.0-3.amzn2023.0.1       amazonlinux 183 k
 libXtst                 x86_64        1.2.5-1.amzn2023.0.1       amazonlinux  22 k
 libbrotli               x86_64        1.0.9-4.amzn2023.0.2       amazonlinux 315 k
 libjpeg-turbo           x86_64        2.1.4-2.amzn2023.0.5       amazonlinux 190 k
 libpng                  x86_64        2:1.6.37-10.amzn2023.0.6   amazonlinux 128 k
 libxcb                  x86_64        1.17.0-1.amzn2023.0.1      amazonlinux 235 k
```

Java is installed.



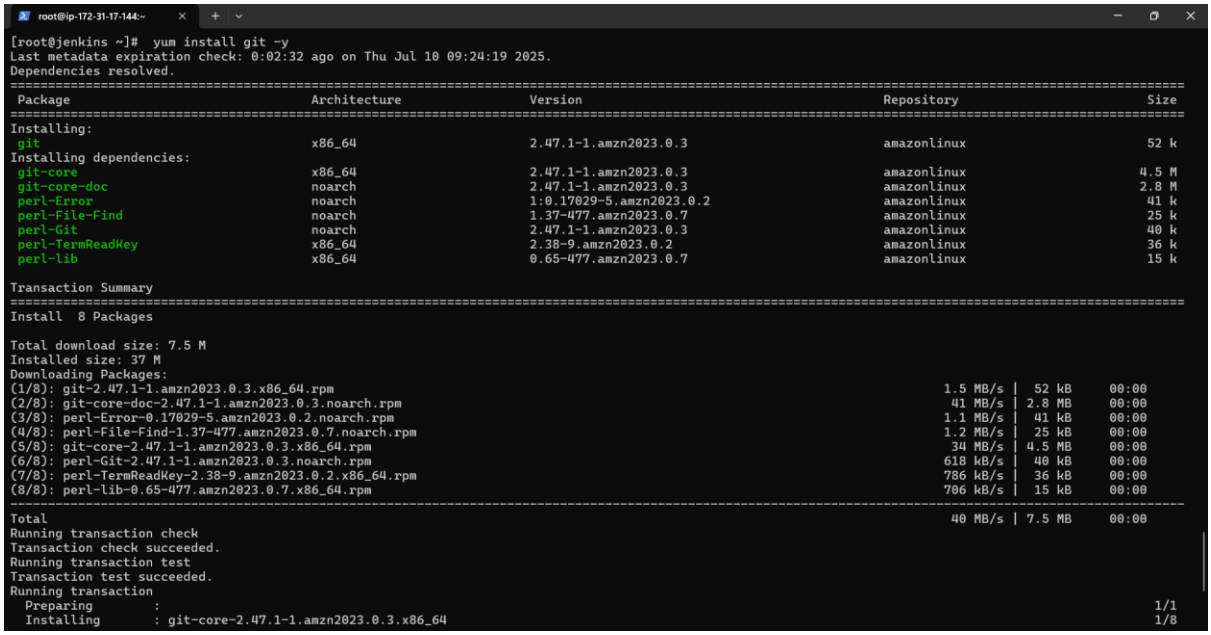
```
root@ip-172-31-17-144:~
[root@jenkins ~]#  yum install jenkins -y
Last metadata expiration check: 0:01:15 ago on Thu Jul 10 09:24:19 2025.
Dependencies resolved.
================================================================================
 Package                 Architecture  Version                    Repository  Size
================================================================================
Installing:
 jenkins                 noarch        2.504.3-1.1                jenkins     90 M

Transaction Summary
================================================================================
Install  1 Package

Total download size: 90 M
Installed size: 90 M
Downloading Packages:
jenkins-2.504.3-1.1.noarch.rpm                  9.4 MB/s |  90 MB     00:09
--------------------------------------------------------------------------------
Total                                           9.4 MB/s |  90 MB     00:09
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Running scriptlet: jenkins-2.504.3-1.1.noarch                             1/1
  Installing       : jenkins-2.504.3-1.1.noarch                             1/1
  Running scriptlet: jenkins-2.504.3-1.1.noarch                             1/1
  Verifying        : jenkins-2.504.3-1.1.noarch                             1/1

Installed:
  jenkins-2.504.3-1.1.noarch

Complete!
[root@jenkins ~]#
```

Jenkins is installed.

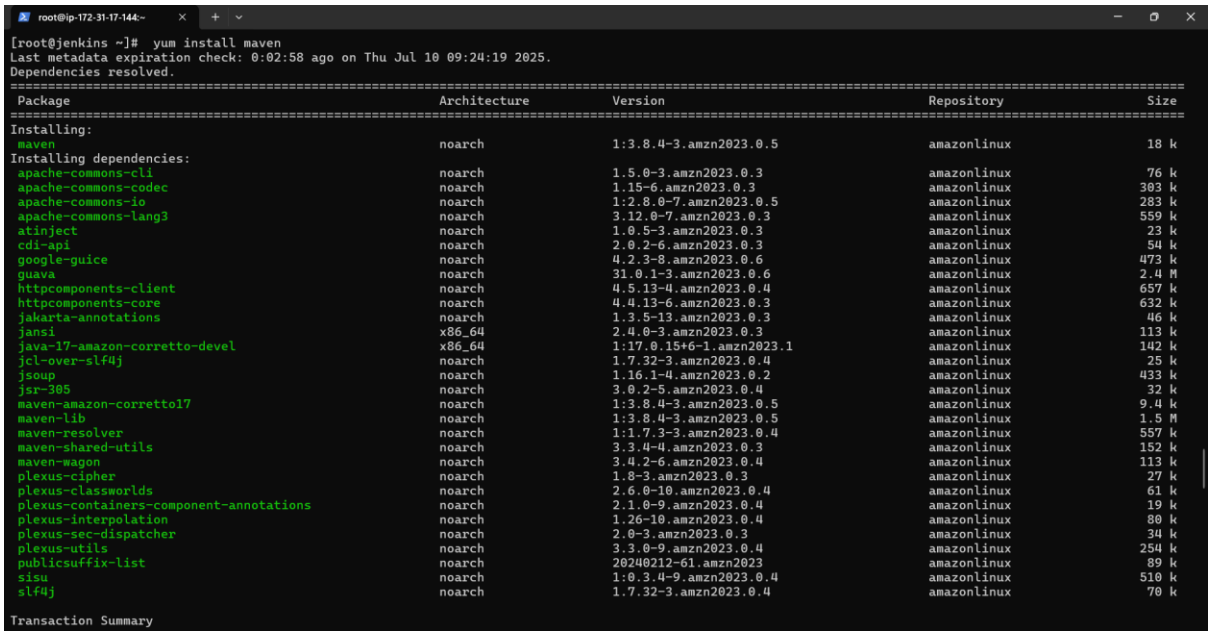


```
root@ip-172-31-17-144:~
[root@jenkins ~]#  systemctl enable jenkins
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
[root@jenkins ~]# systemctl start jenkins
[root@jenkins ~]# systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: disabled)
     Active: active (running) since Thu 2025-07-10 09:26:27 UTC; 2s ago
   Main PID: 26271 (java)
      Tasks: 46 (limit: 4656)
     Memory: 709.7M
        CPU: 18.549s
     CGroup: /system.slice/jenkins.service
             └─26271 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Jul 10 09:26:23 jenkins.example.com jenkins[26271]: e502753074cf4495ae0f7309e0400bbb
Jul 10 09:26:23 jenkins.example.com jenkins[26271]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jul 10 09:26:23 jenkins.example.com jenkins[26271]: *************************************************************
Jul 10 09:26:23 jenkins.example.com jenkins[26271]: *************************************************************
Jul 10 09:26:23 jenkins.example.com jenkins[26271]: *************************************************************
Jul 10 09:26:27 jenkins.example.com jenkins[26271]: 2025-07-10 09:26:27.409+0000 [id=30]        INFO        jenkins.InitReactorRunner$1#onAttained: Complet>
Jul 10 09:26:27 jenkins.example.com jenkins[26271]: 2025-07-10 09:26:27.442+0000 [id=23]        INFO        hudson.lifecycle.Lifecycle#onReady: Jenkins is >
Jul 10 09:26:27 jenkins.example.com systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Jul 10 09:26:27 jenkins.example.com jenkins[26271]: 2025-07-10 09:26:27.729+0000 [id=49]        INFO        h.m.DownloadService$Downloadable#load: Obtained>
Jul 10 09:26:27 jenkins.example.com jenkins[26271]: 2025-07-10 09:26:27.731+0000 [id=49]        INFO        hudson.util.Retrier#start: Performed the action>
lines 1-20/20 (END)
```

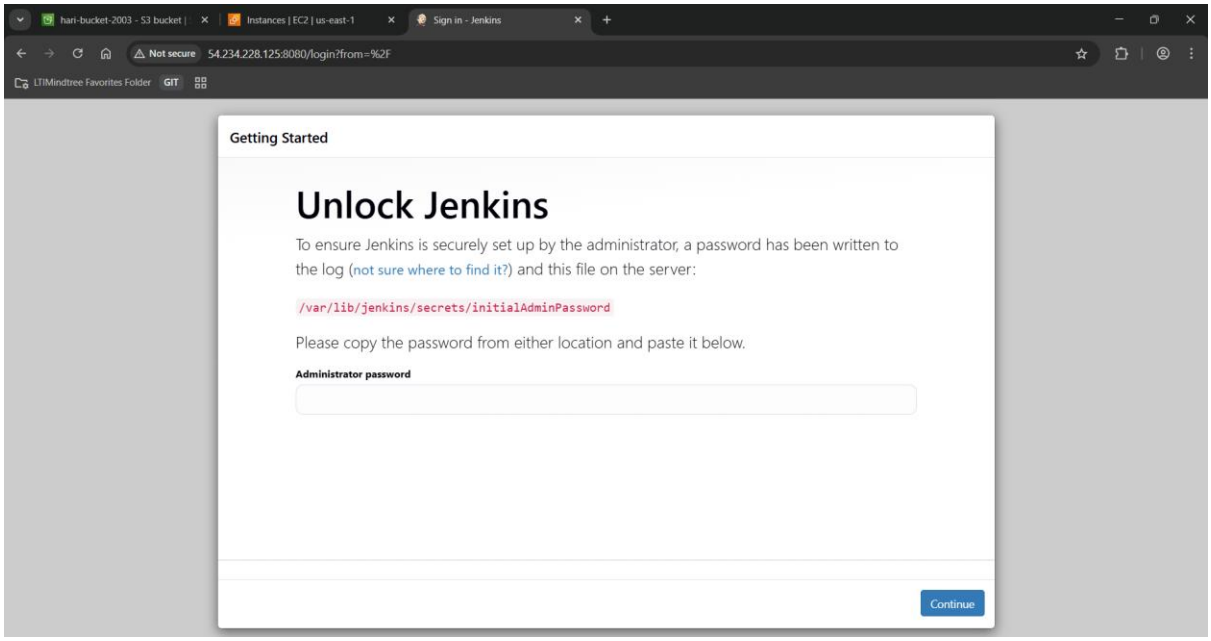
Jenkins is started and enabled. Jenkins status is active.



Git is installed.

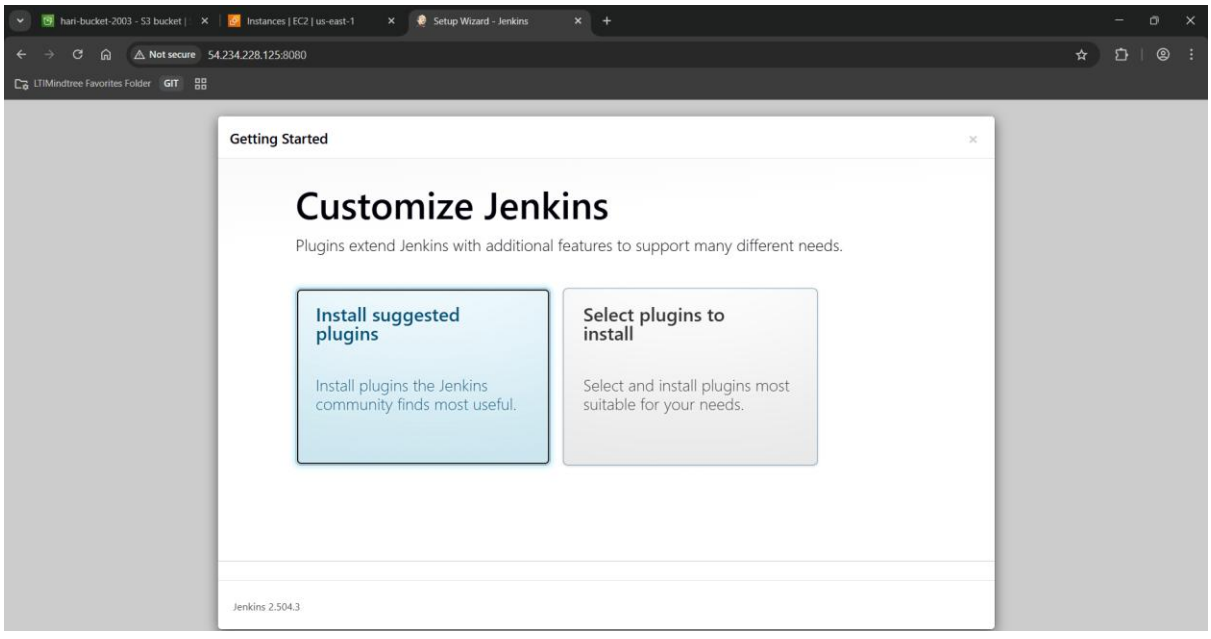


Maven is installed.

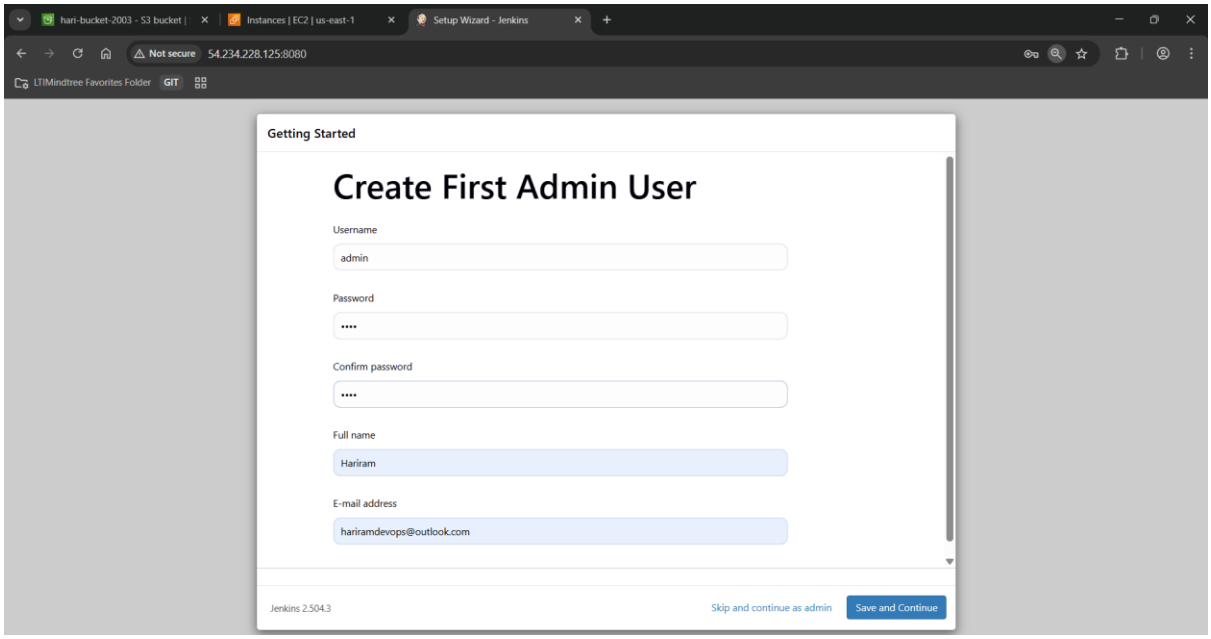


Jenkins dashboard is accessed using instances public ip:8080.

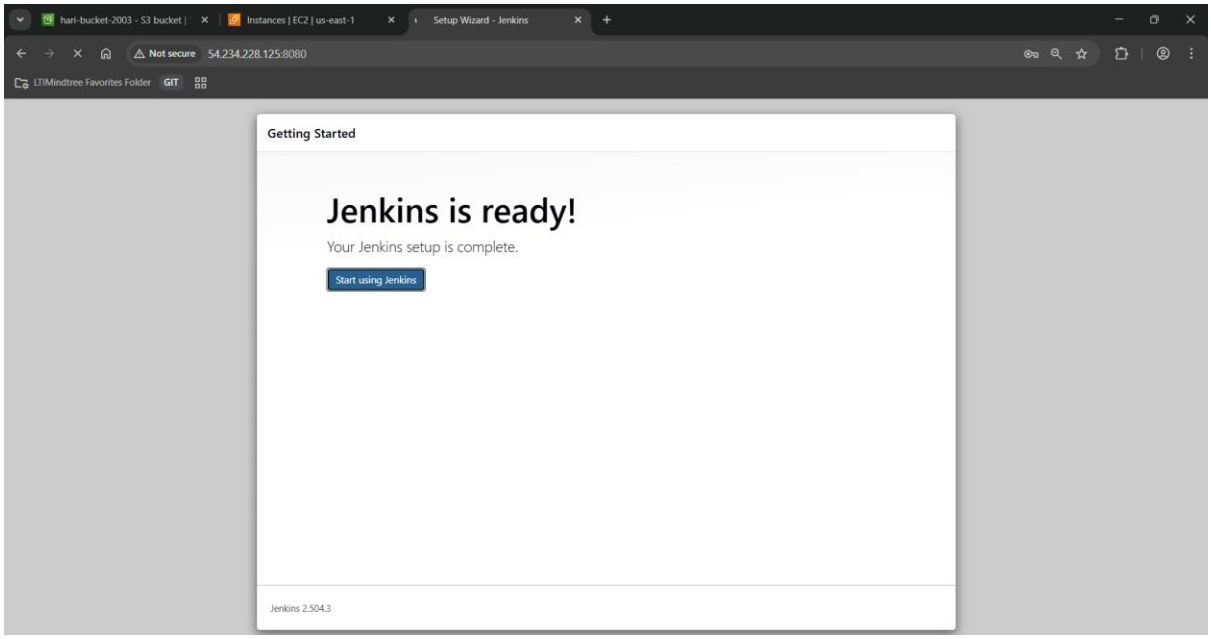
Password is taken from Jenkins terminal.



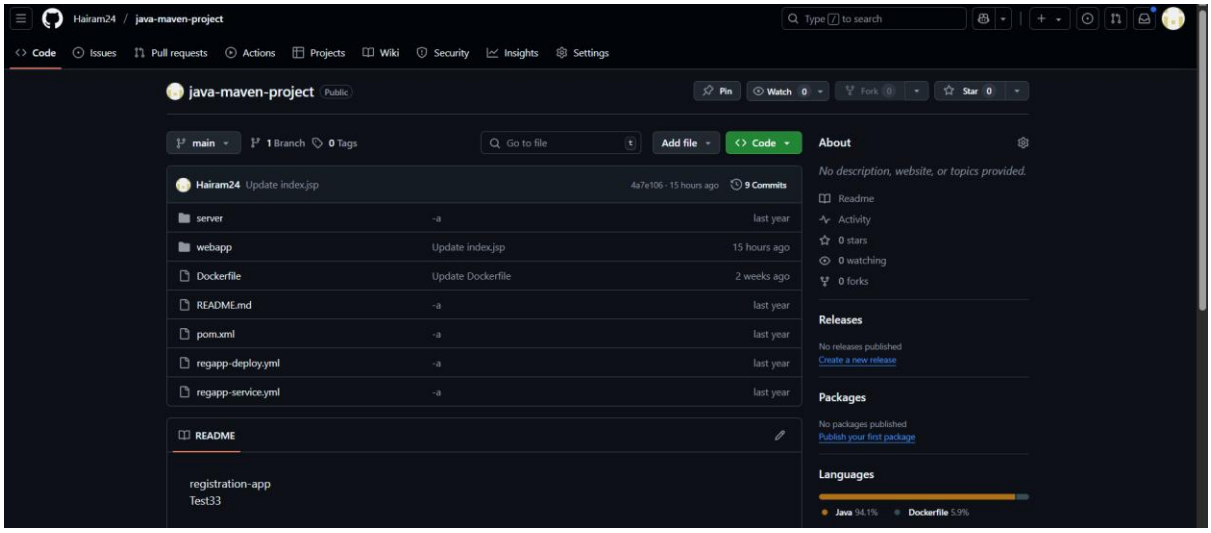
Suggested plugins are installing.



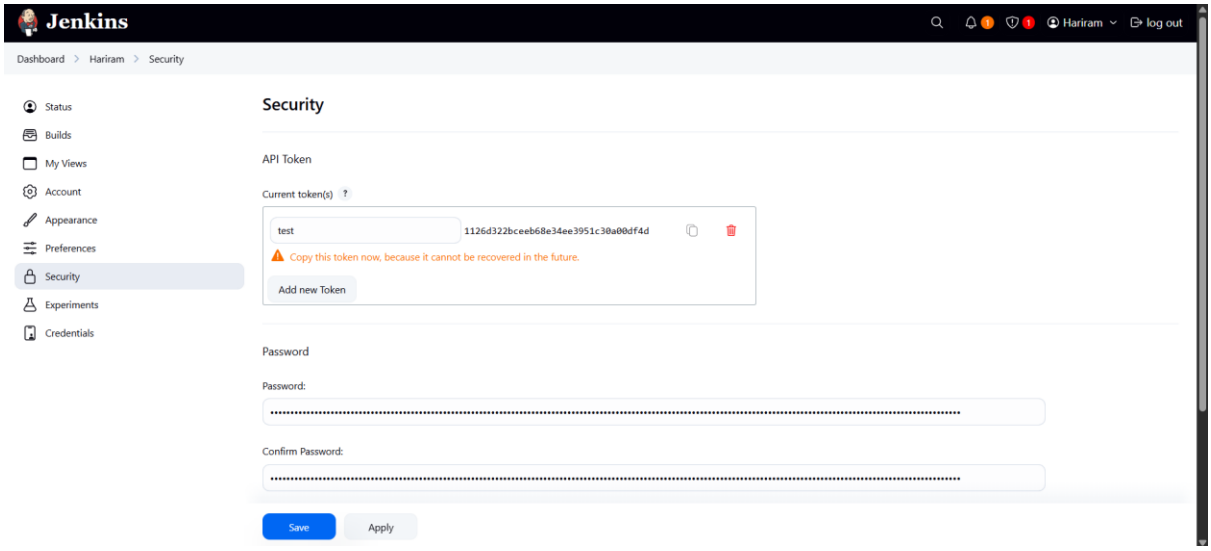
Admin user is created.



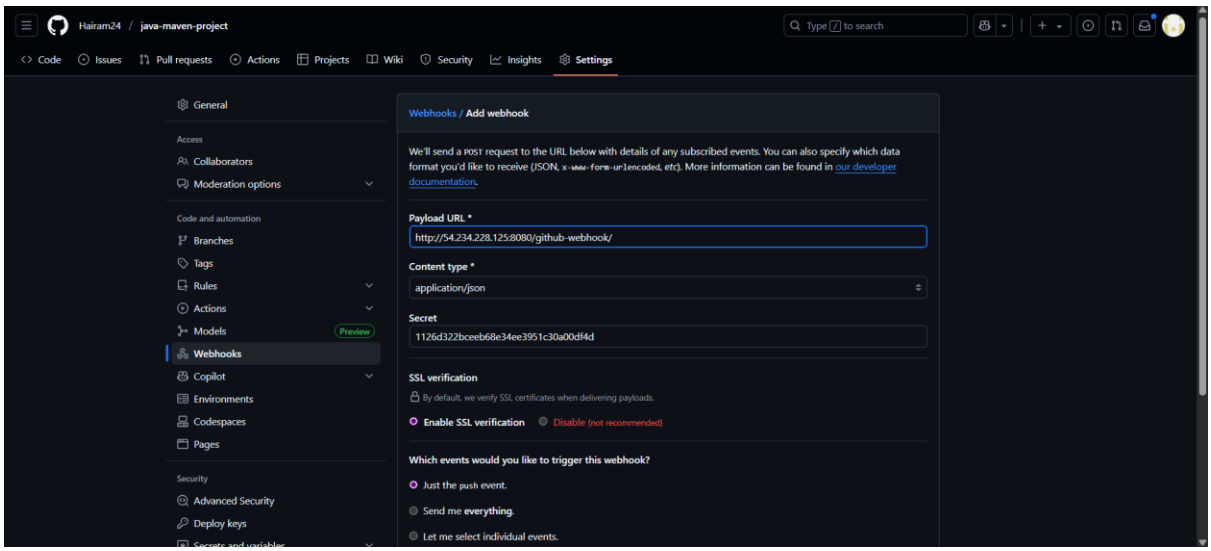
Jenkins is ready to used.



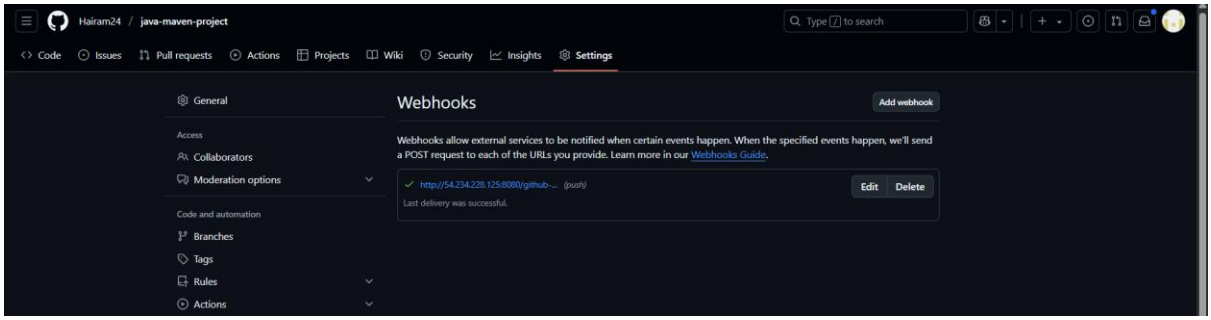
Existing repository is use where maven project is available.



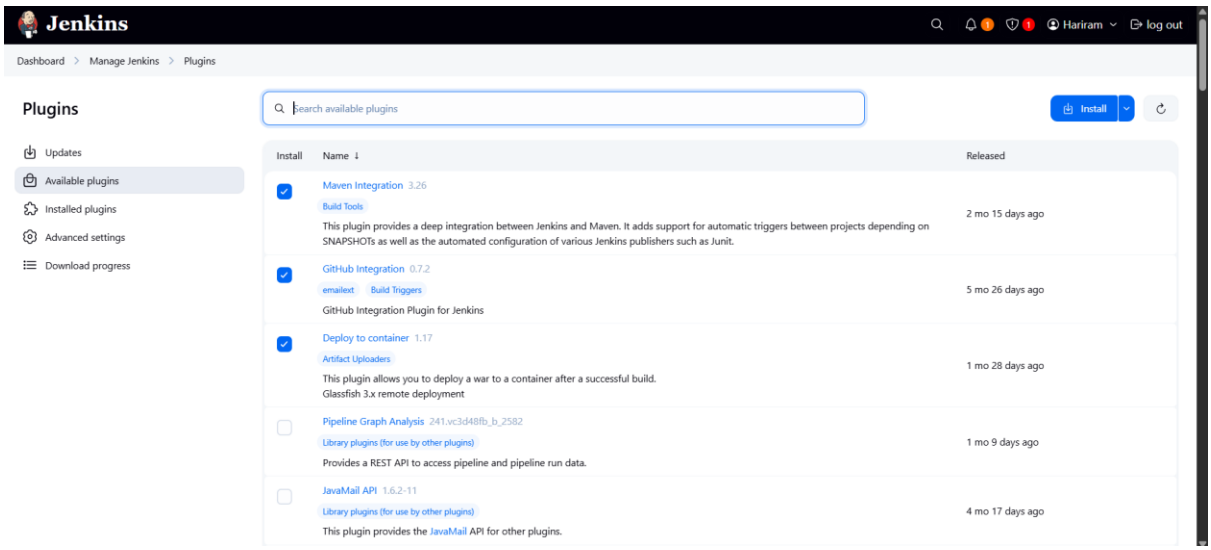
Token is generate in Jenkins to connect it with github.



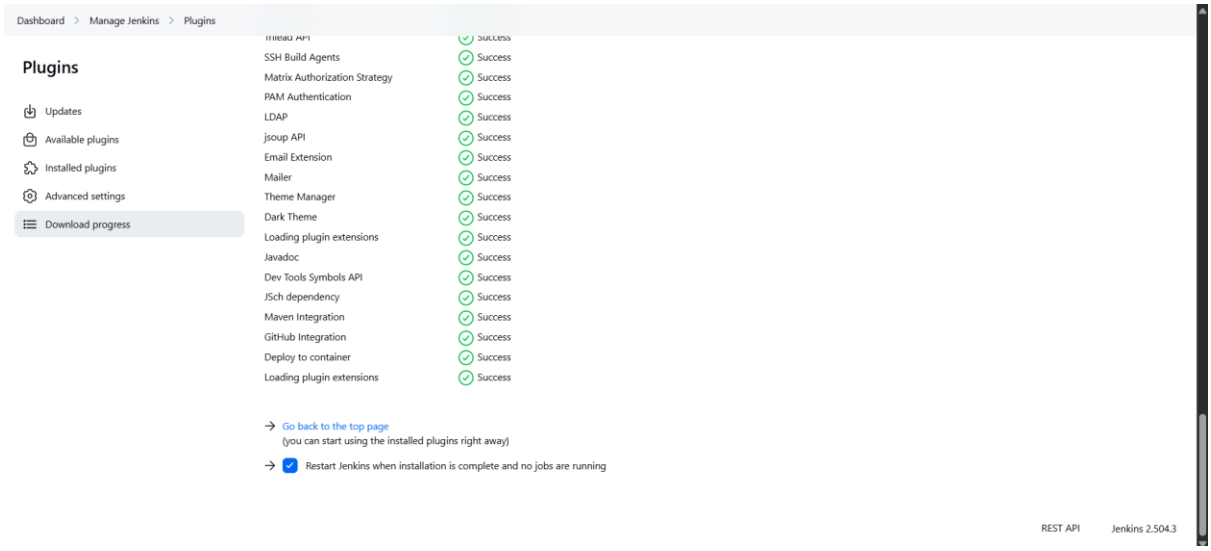
Token is pasted in git webhook and Jenkins url is pasted.



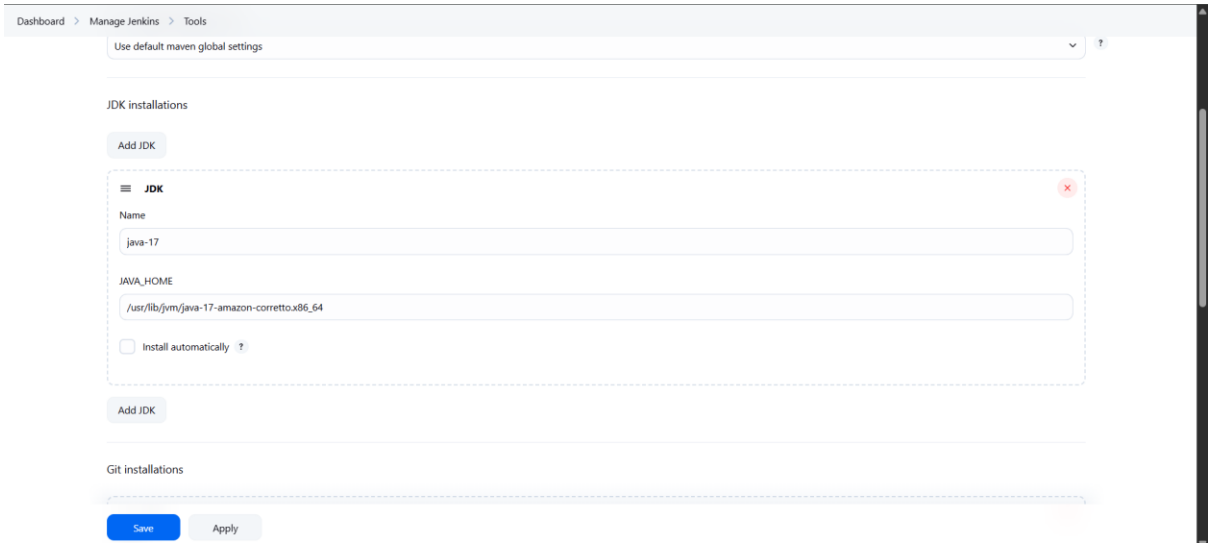
Webhooks is added successfully.



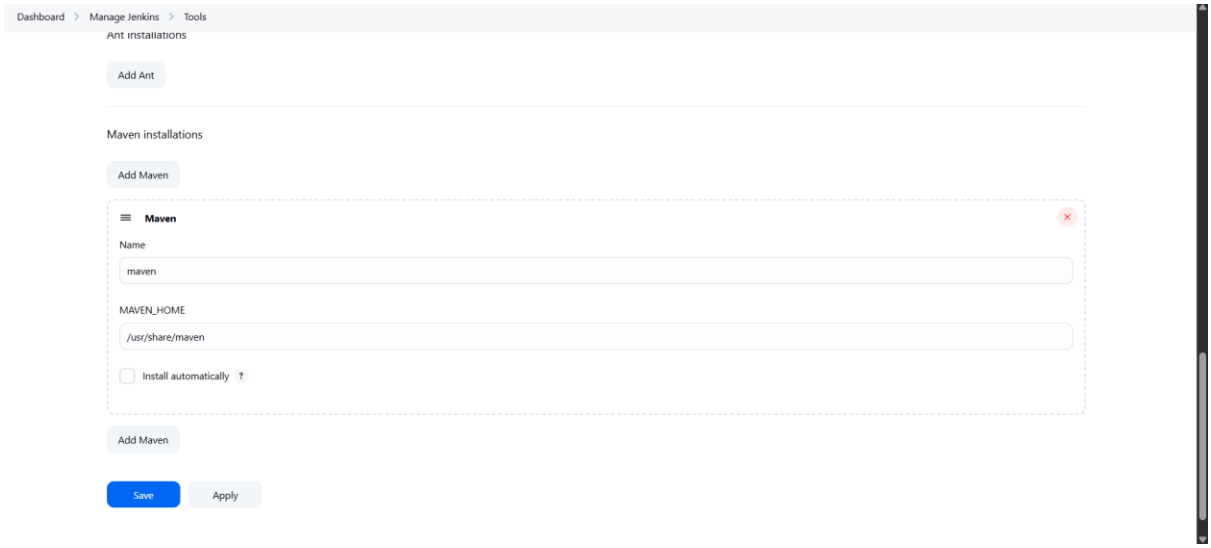
Necessary plugins are added.



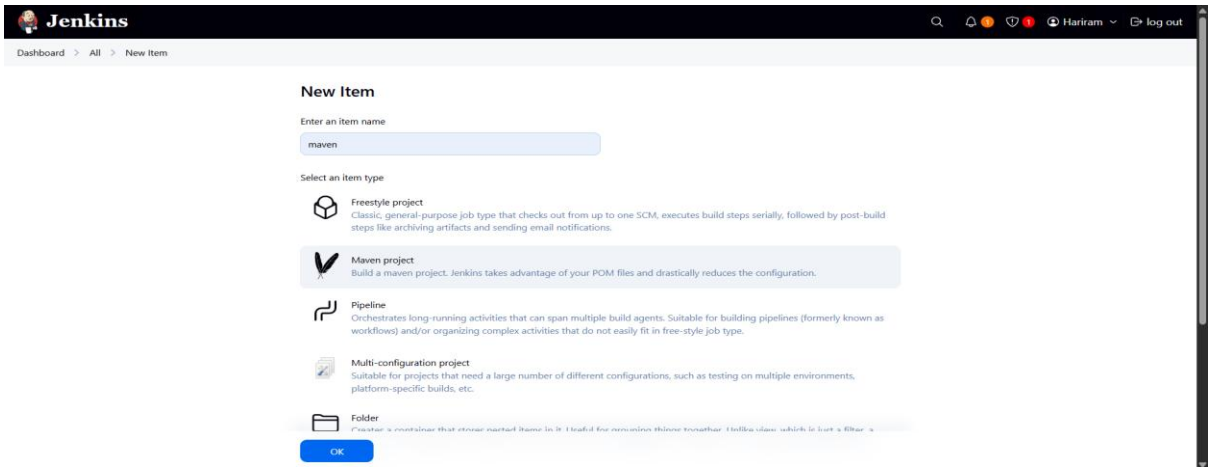
Restarted after installation.



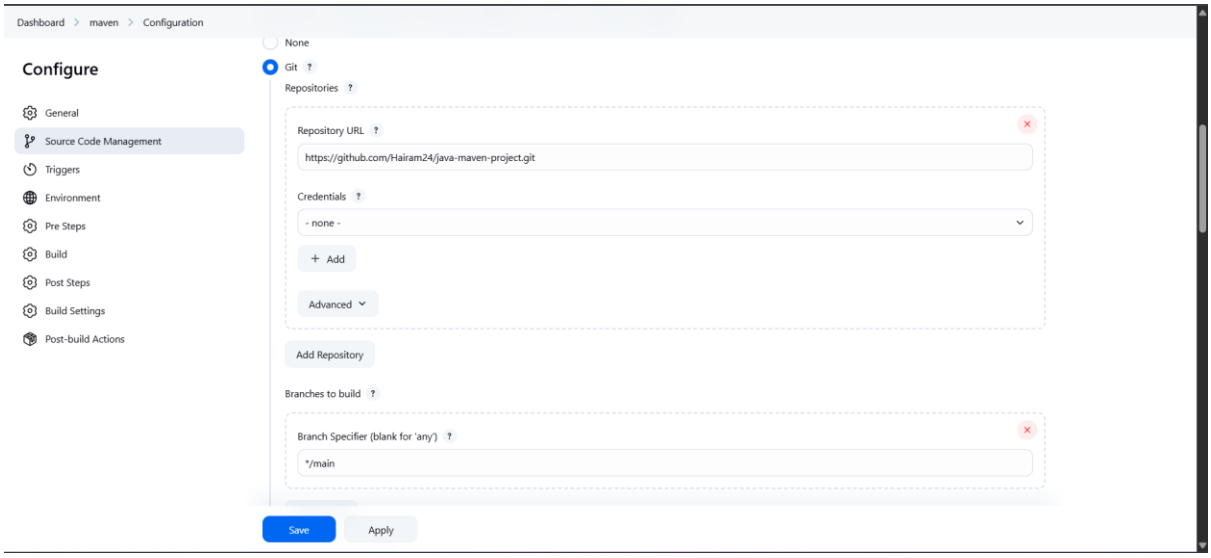
Java path is added from the Jenkins terminal.



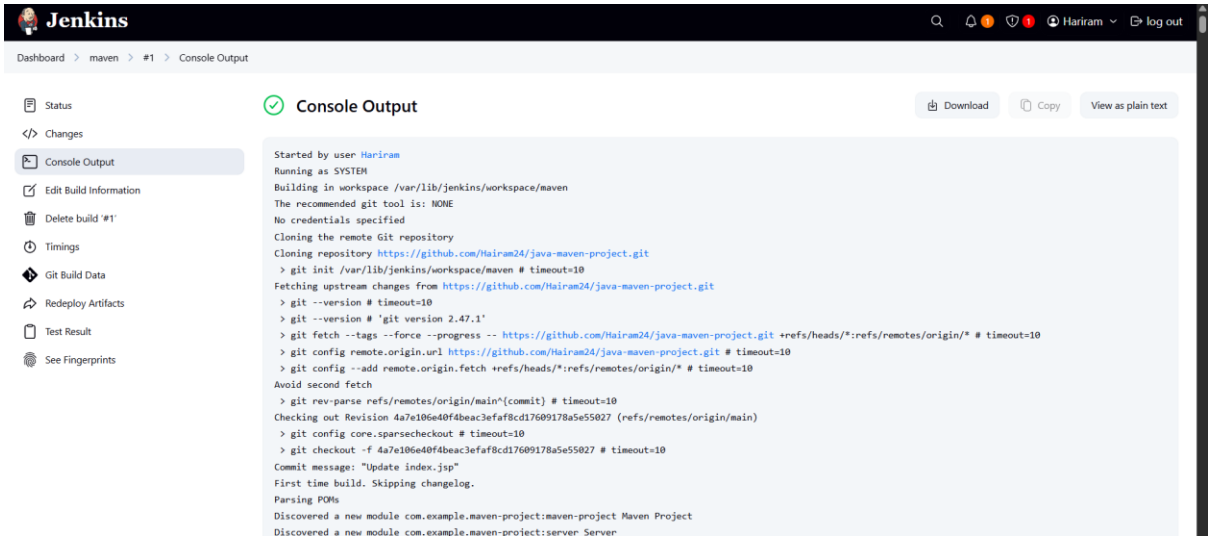
Maven path is added from the Jenkins terminal.



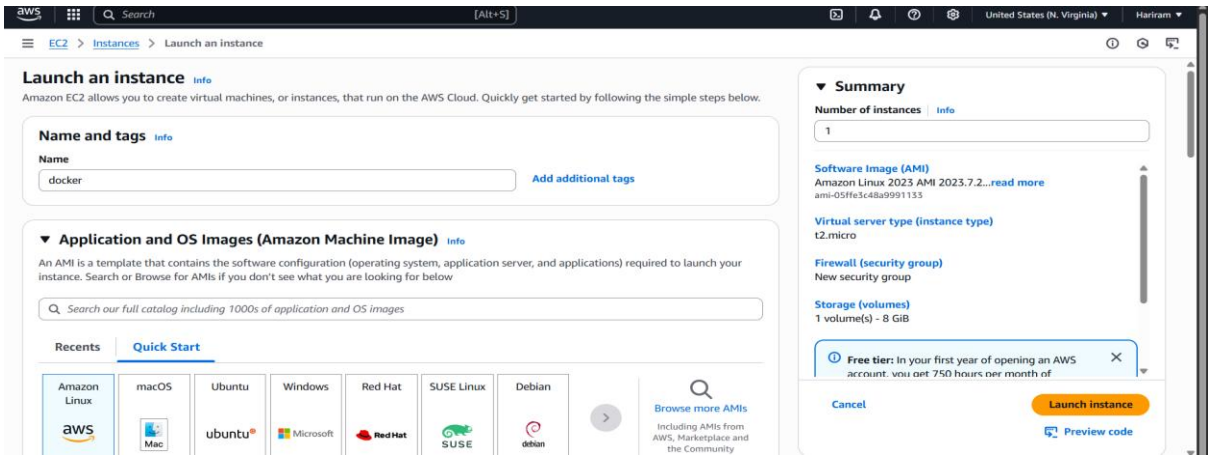
New maven project is created.



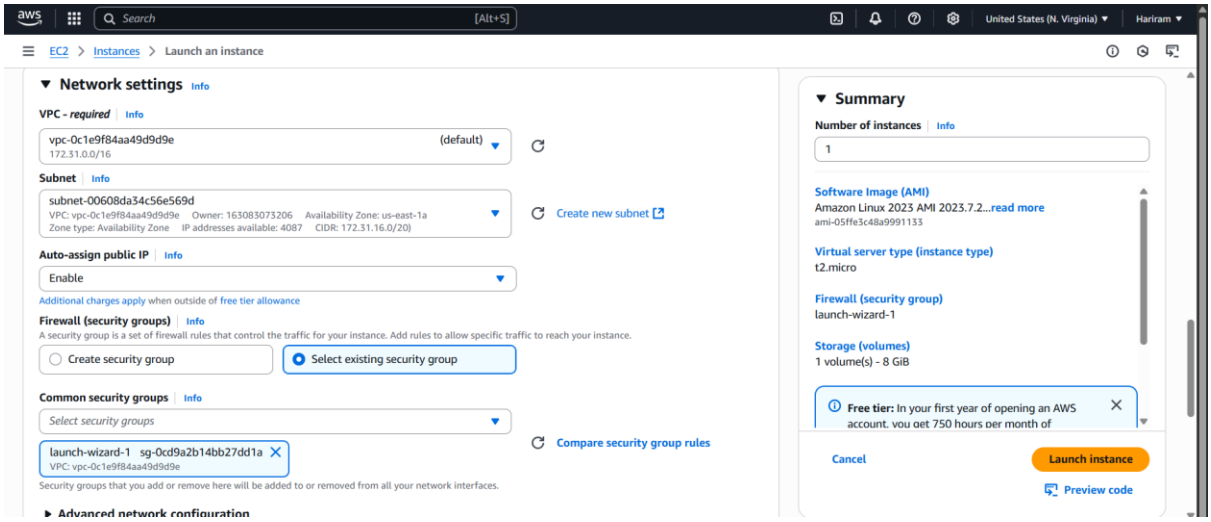
Git http link is added and /master is changed to /main.



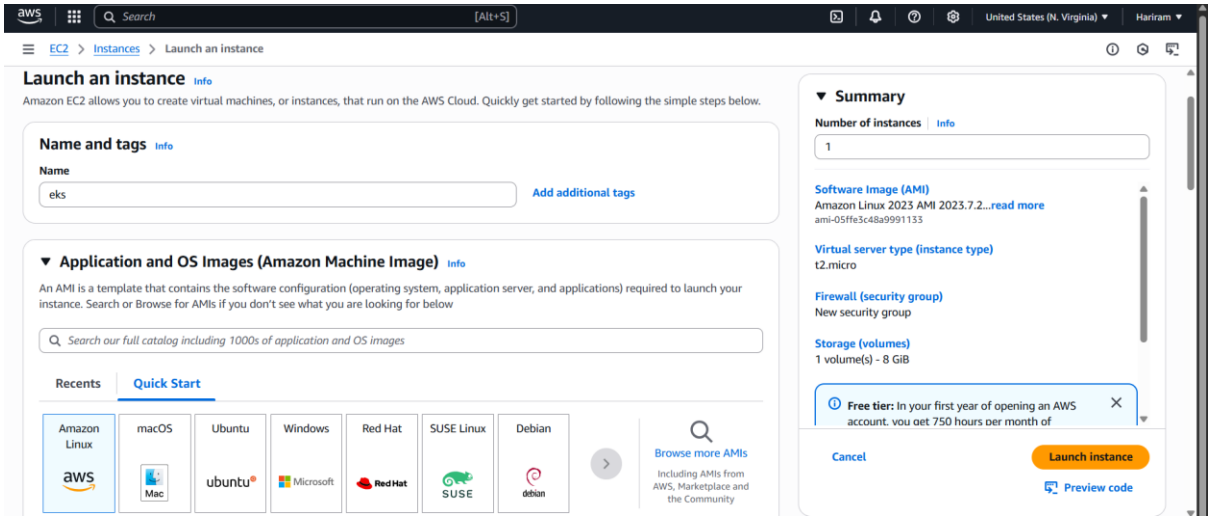
Build is successful and war is created in the Jenkins.



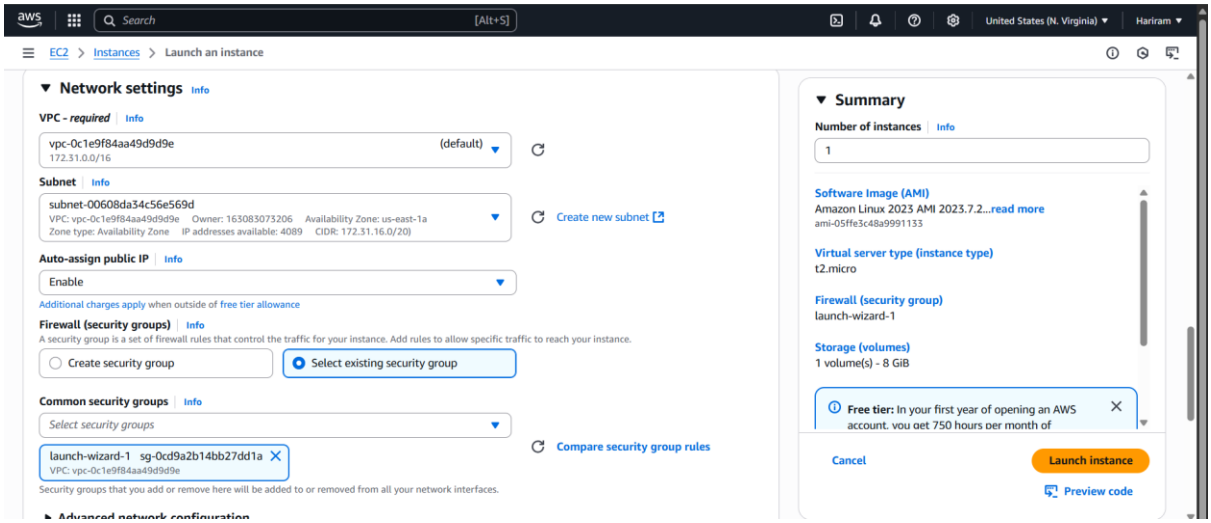
Docker instance is created.



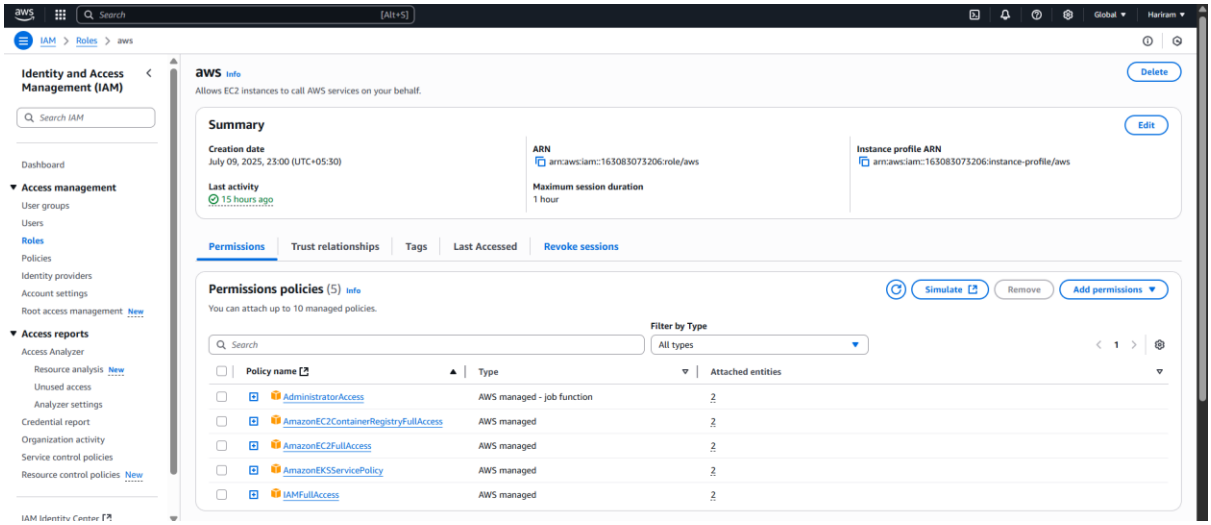
Existing key pair is selected and existing security group is selected.



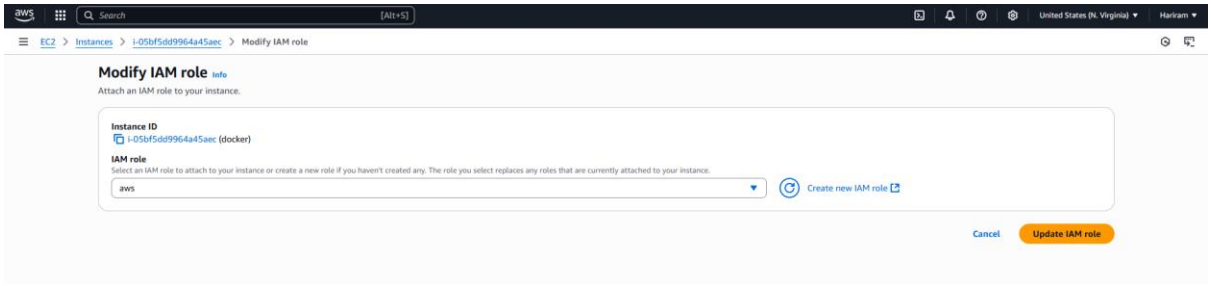
Eks instance is created.



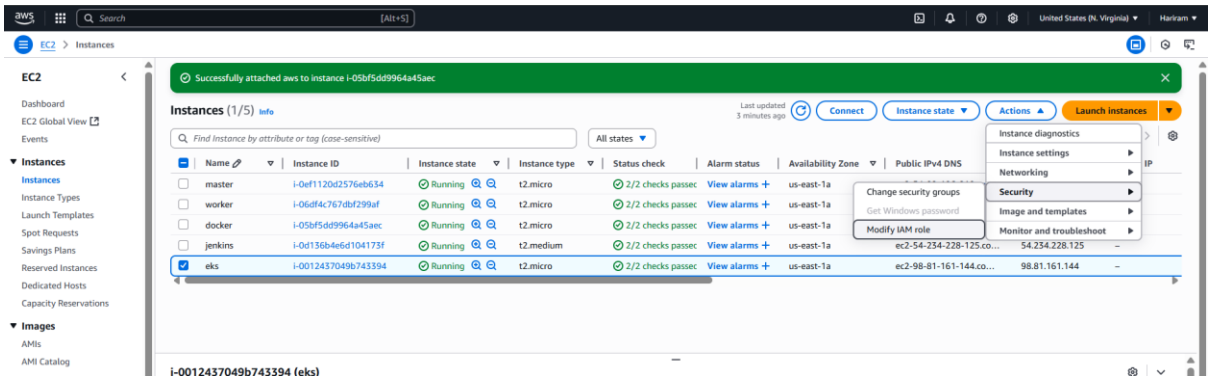
Existing key pair is selected and existing security group is selected.



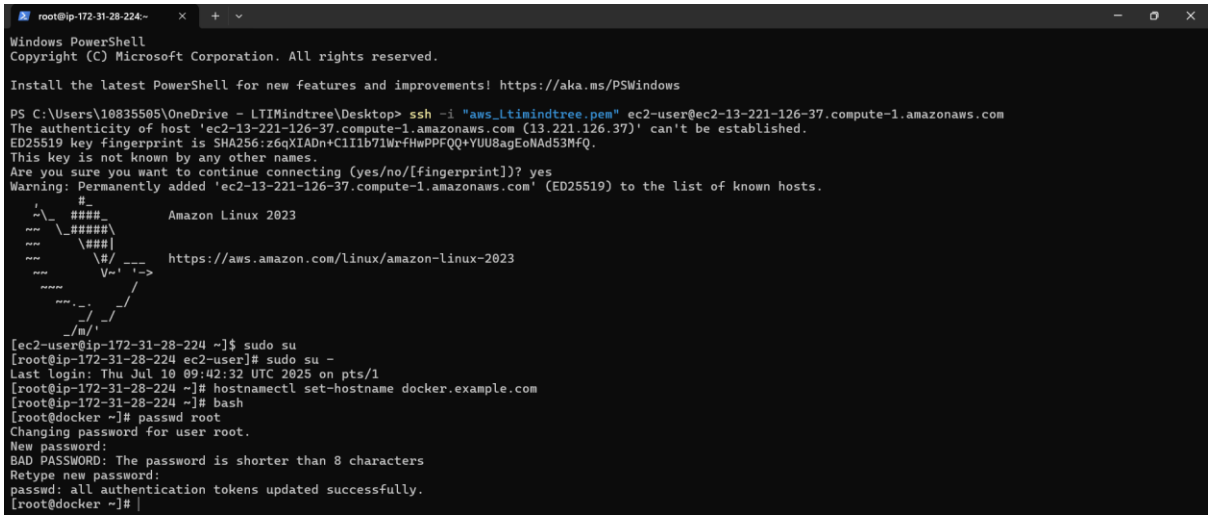
Role is created with necessary permissions.



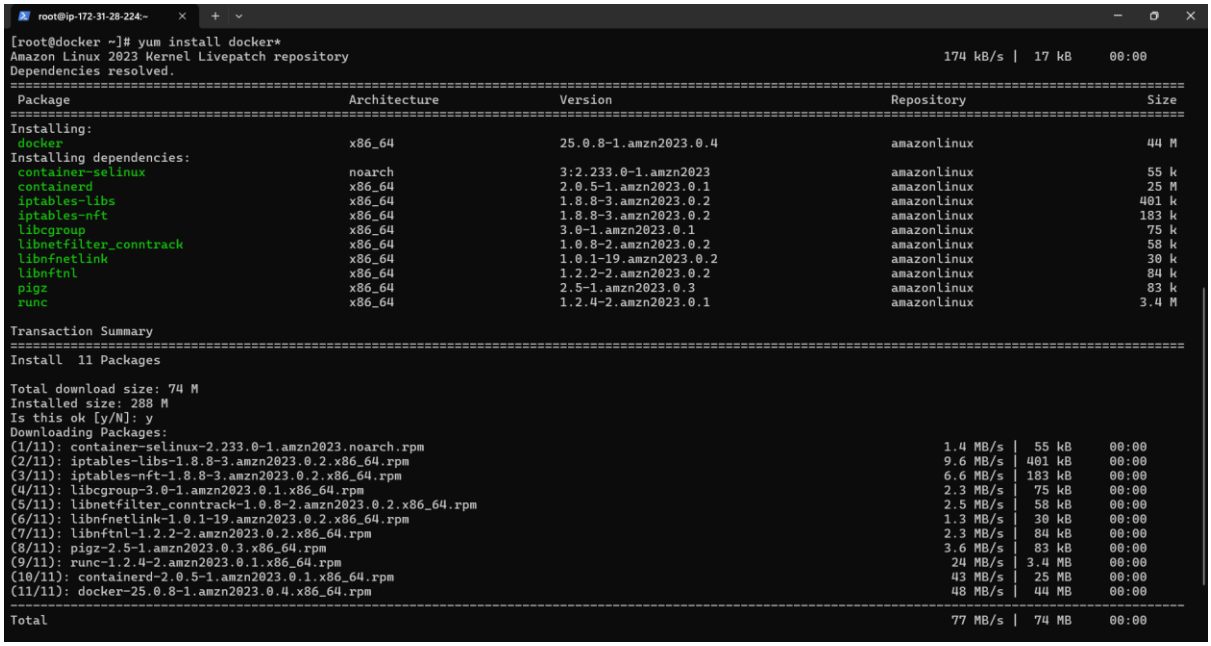
Role Added to docker instance.



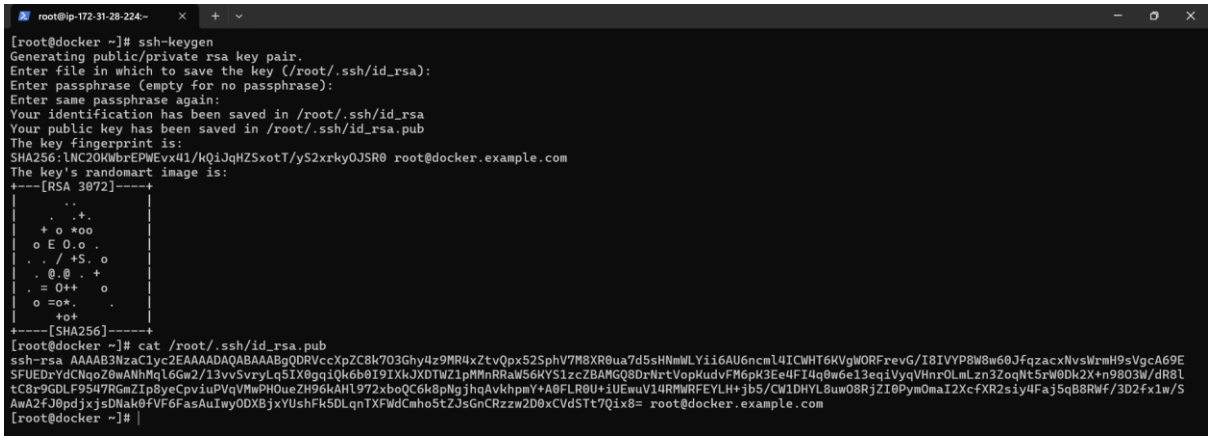
Role added to eks instance.



Docker instance is accessible,

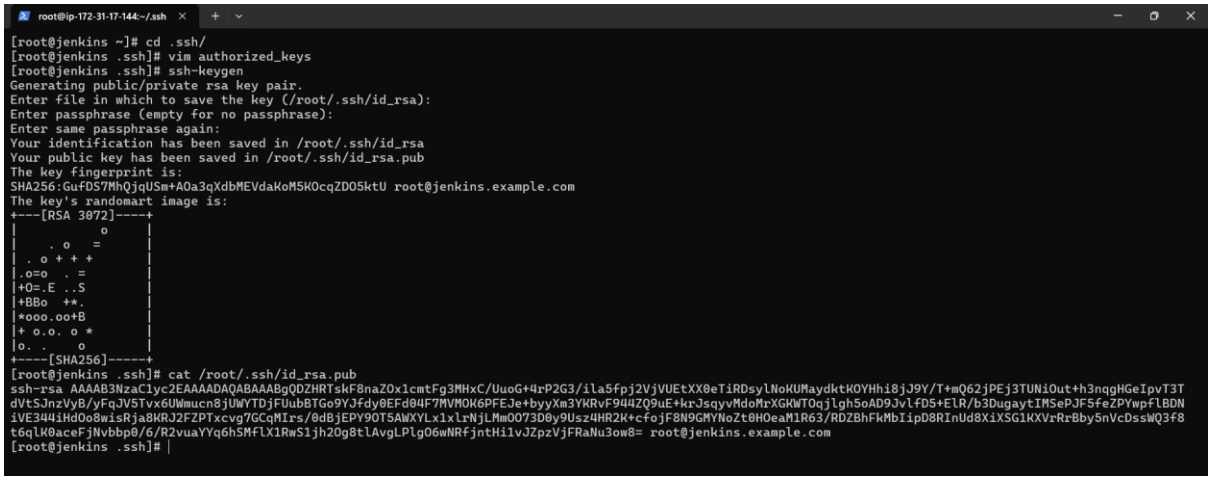


Docker is installed.



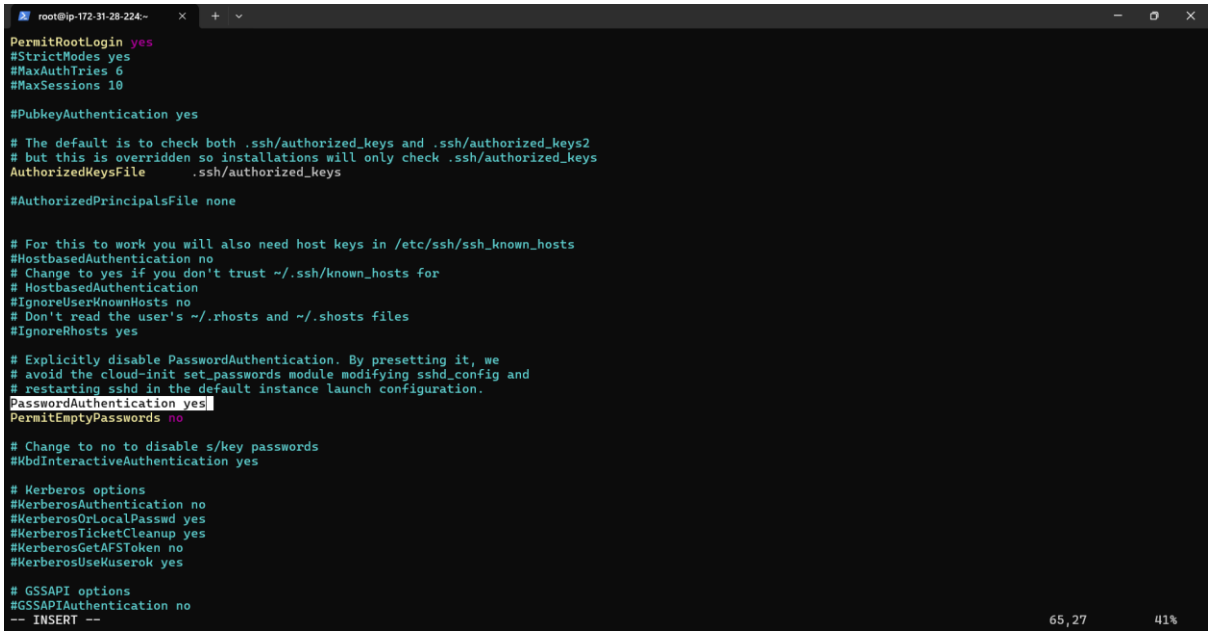
Ssh-keygen is done in docker.

Docker pub key is added to the Jenkins.



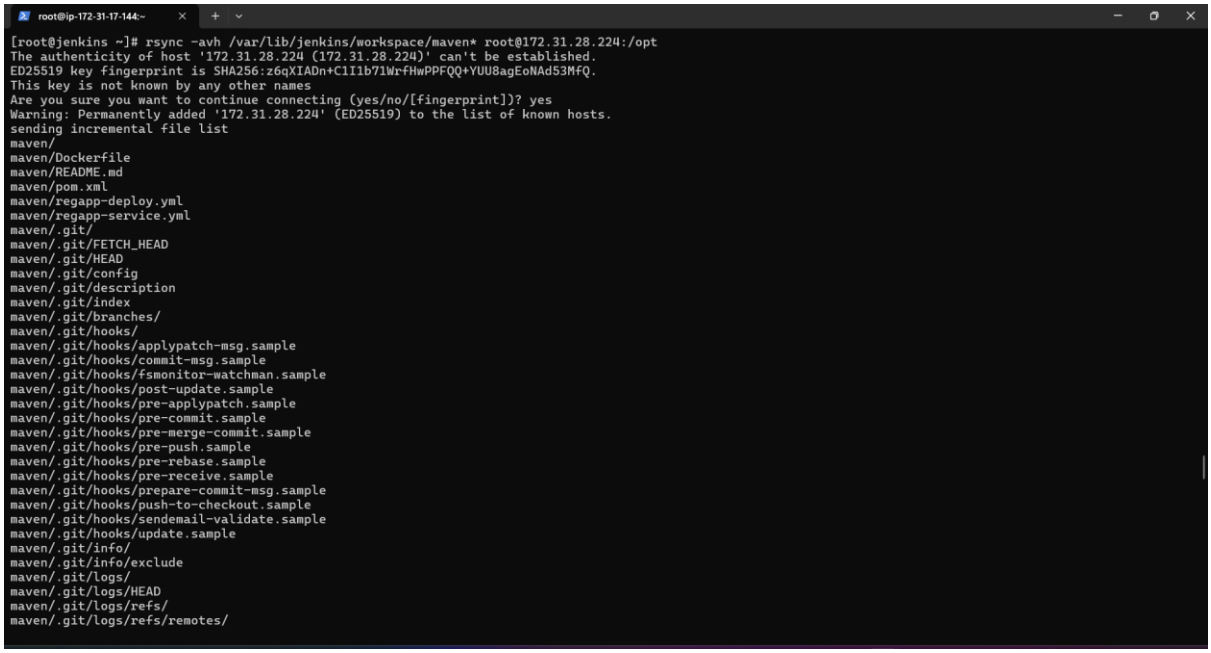
ssh-keygen is done in Jenkins.

Jenkins pub key is added to the docker.



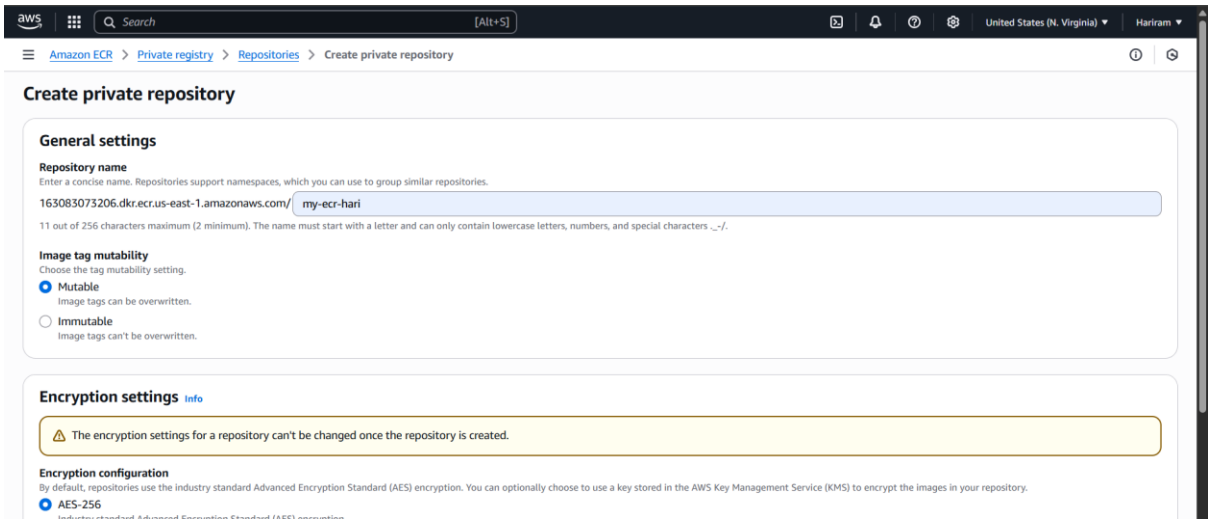
Sshd config is done in docker with password auth and permit login enabled.

Sshd is restarted and enabled.

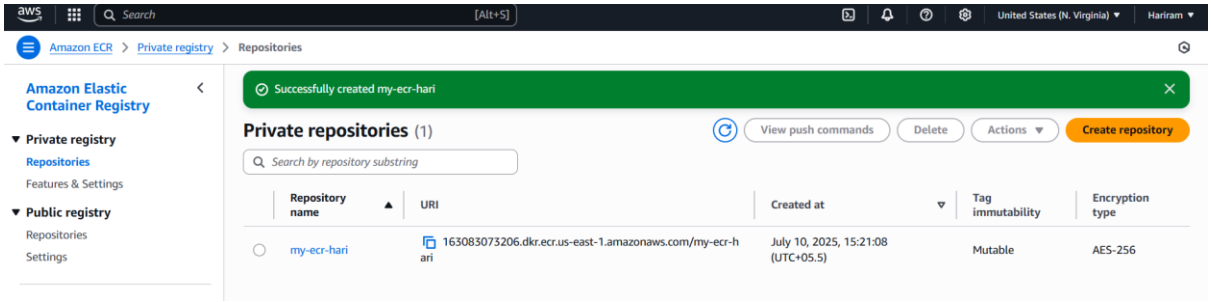


Using rsync cmd build maven project with war file is copied from Jenkins to the docker /opt folder.

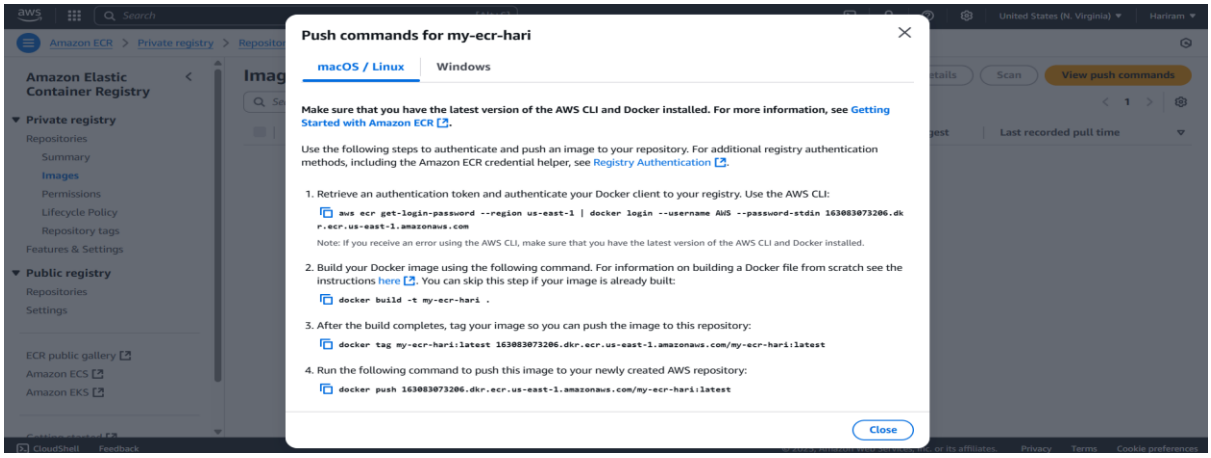
Files successfully copied to /opt of docker.



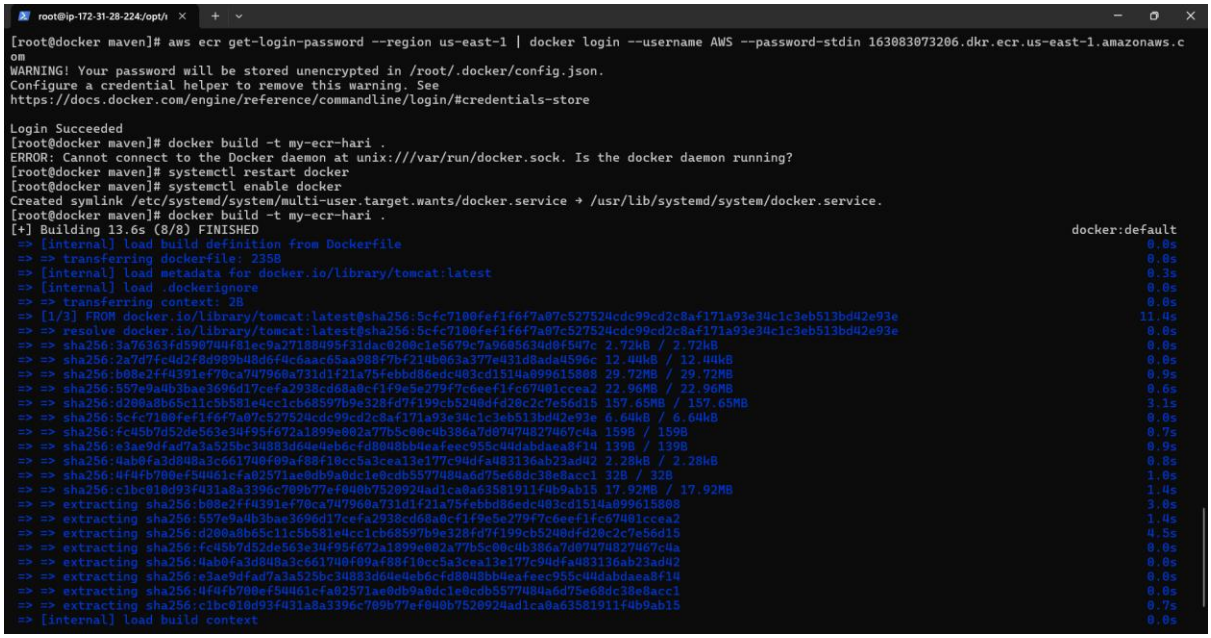
ECR is created.



Ecr created successfully.



Push commands are taken from ecr.



Push cmd is pasted and successfully logged in.

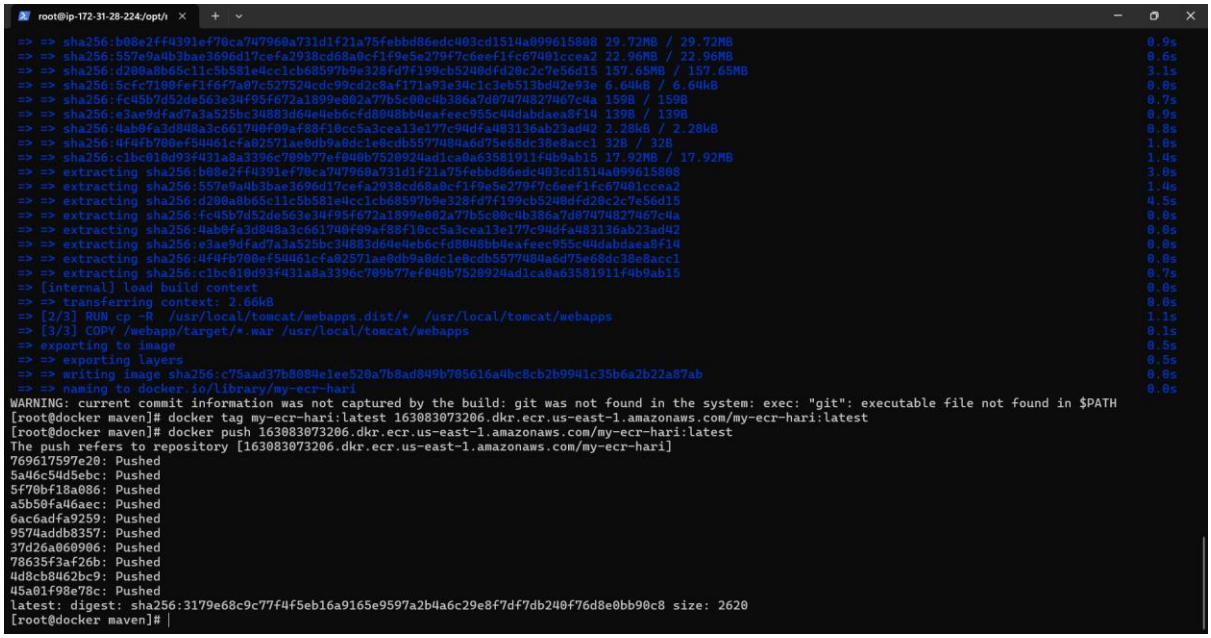
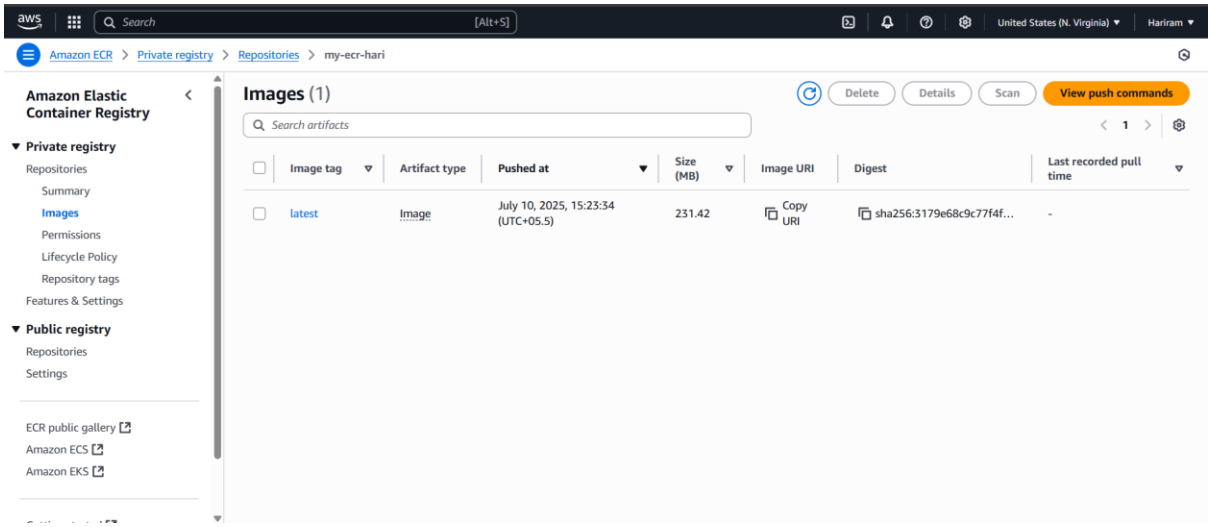
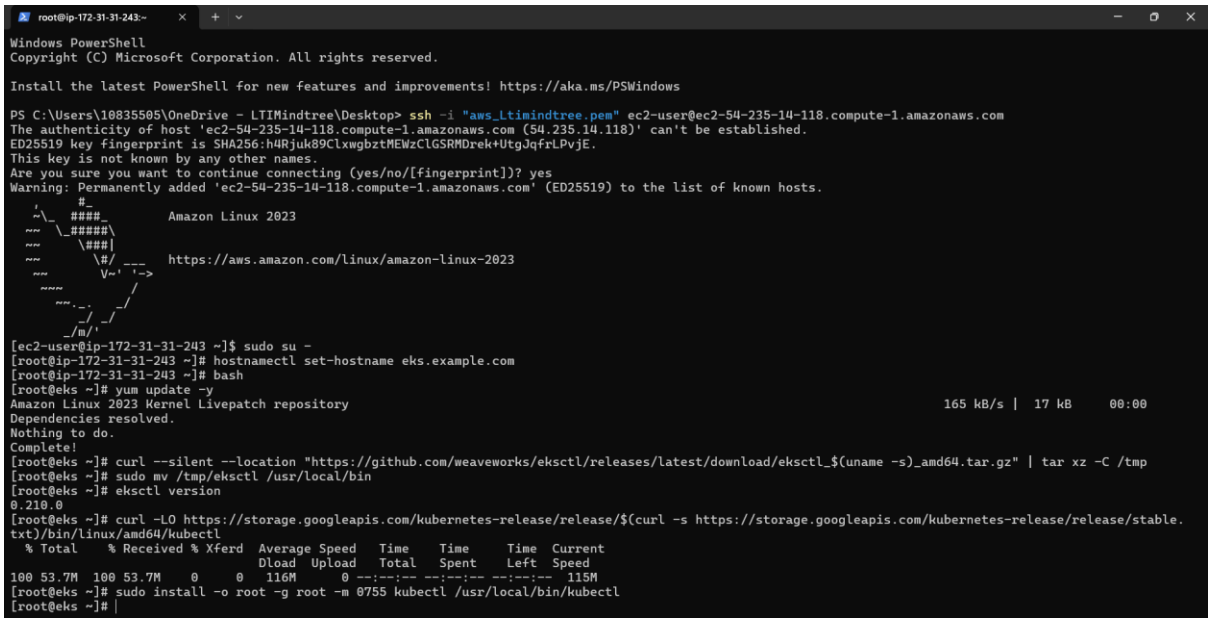


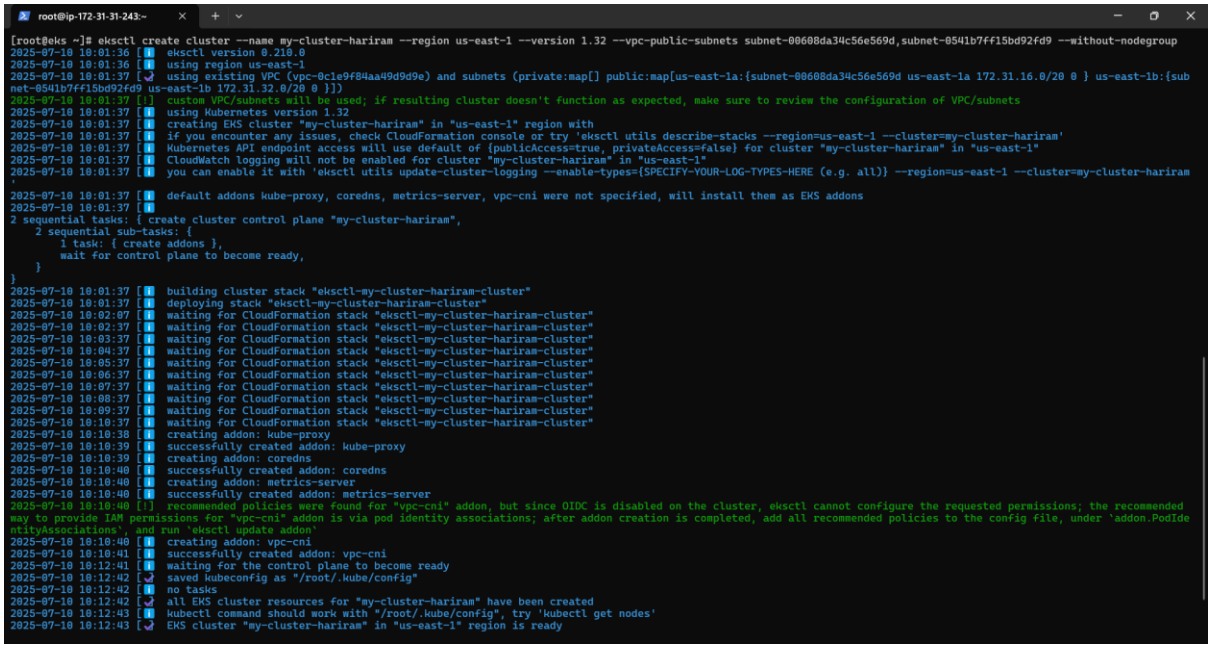
Image is created and pushed into the ECR.



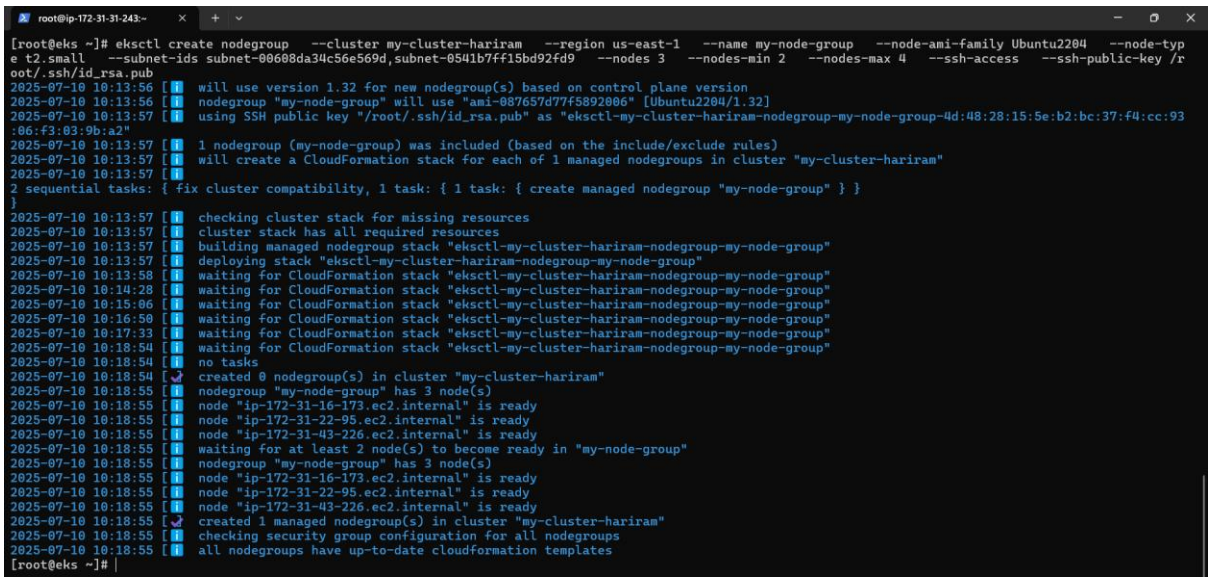
Latest image is available.



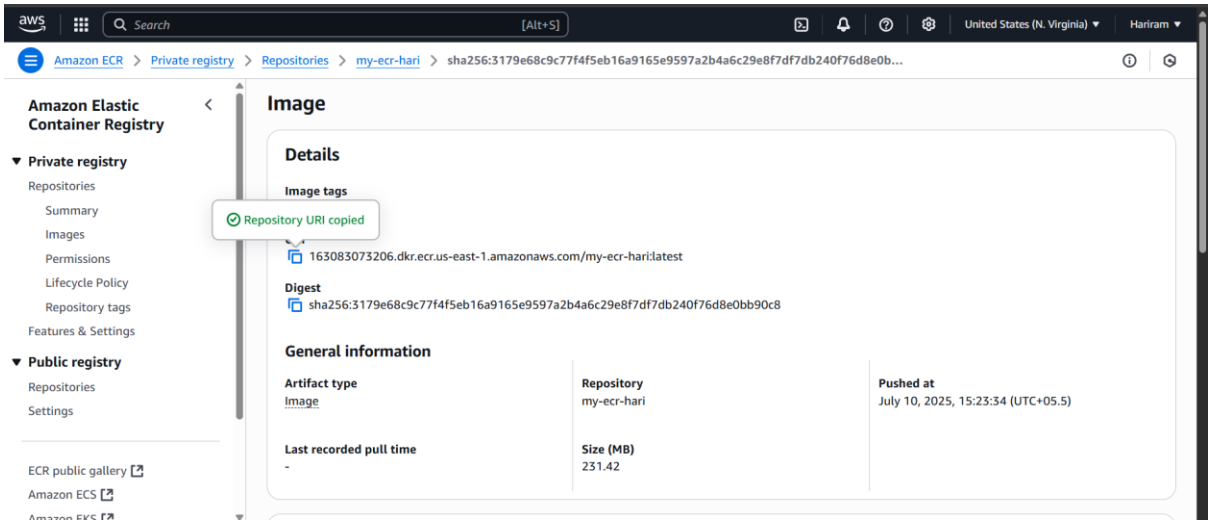
Eks instance is accessible. Necessary dependencies are installed.



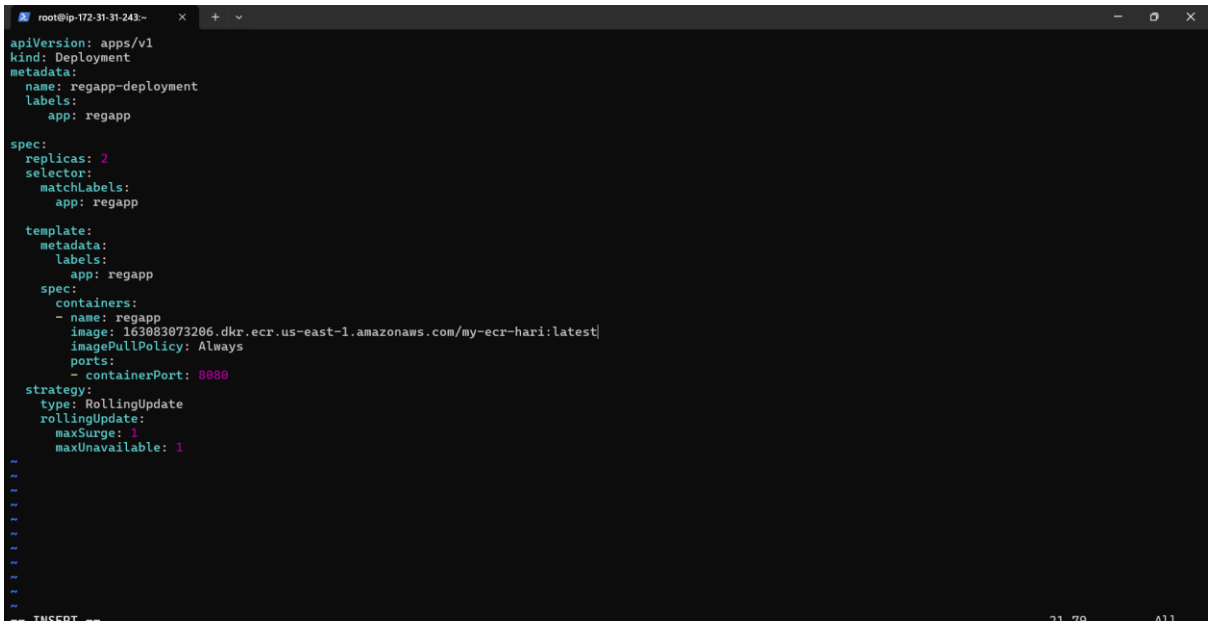
Cluster is created.



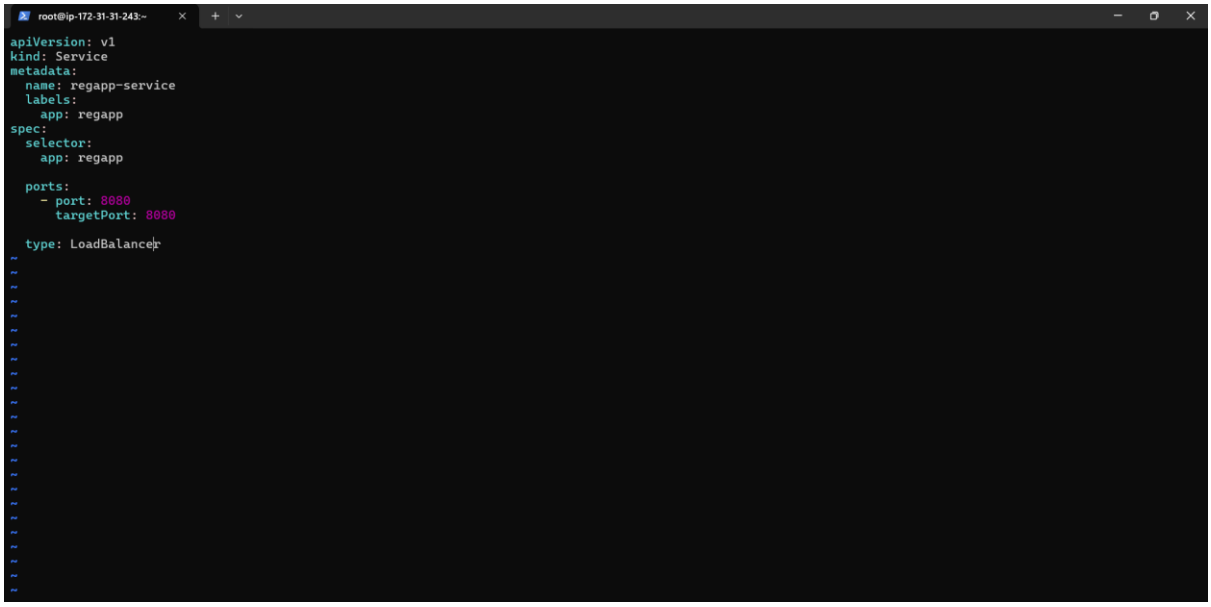
Node group is created.



URI of the latest image is copied.

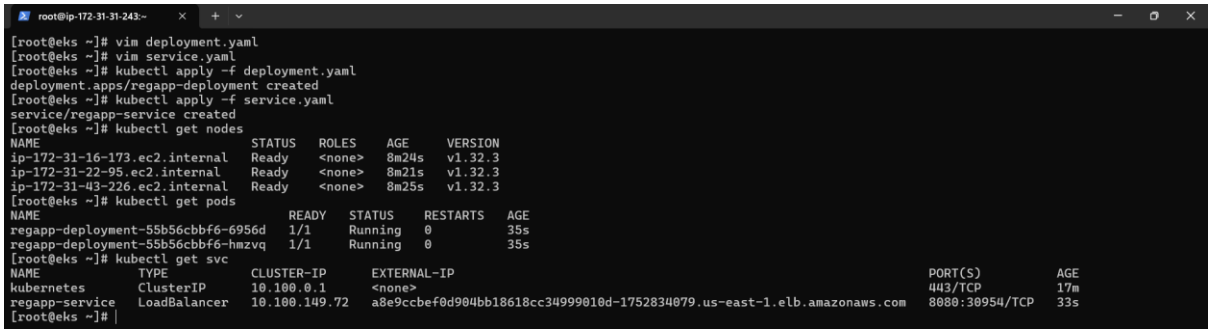


Deployment.yaml file is added with lates URI of the image.



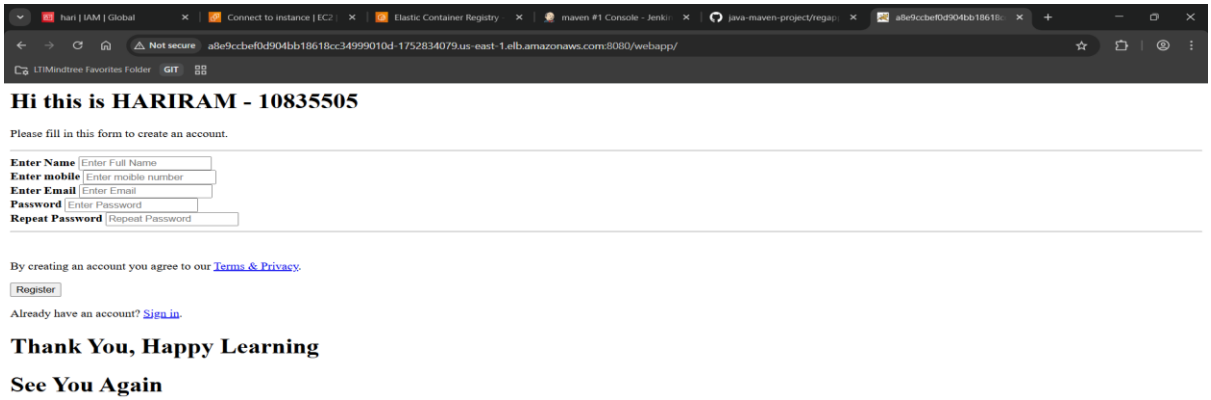
```
root@ip-172-31-31-243:~
apiVersion: v1
kind: Service
metadata:
  name: regapp-service
  labels:
    app: regapp
spec:
  selector:
    app: regapp

  ports:
    - port: 8080
      targetPort: 8080

  type: LoadBalancer
```

Service.yaml file is added with port enabled.



```
root@ip-172-31-31-243:~
[root@eks ~]# vim deployment.yaml
[root@eks ~]# vim service.yaml
[root@eks ~]# kubectl apply -f deployment.yaml
deployment.apps/regapp-deployment created
[root@eks ~]# kubectl apply -f service.yaml
service/regapp-service created
[root@eks ~]# kubectl get nodes
NAME                          STATUS   ROLES    AGE     VERSION
ip-172-31-16-173.ec2.internal   Ready    <none>   8m24s   v1.32.3
ip-172-31-22-95.ec2.internal    Ready    <none>   8m21s   v1.32.3
ip-172-31-43-226.ec2.internal   Ready    <none>   8m25s   v1.32.3
[root@eks ~]# kubectl get pods
NAME                                 READY   STATUS    RESTARTS   AGE
regapp-deployment-55b56cbbf6-6956d   1/1     Running   0          35s
regapp-deployment-55b56cbbf6-hmzvq   1/1     Running   0          35s
[root@eks ~]# kubectl get svc
NAME             TYPE           CLUSTER-IP      EXTERNAL-IP                                                              PORT(S)          AGE
kubernetes       ClusterIP      10.100.0.1      <none>                                                                   443/TCP          17m
regapp-service   LoadBalancer   10.100.149.72   a8e9ccbef0d904bb18618cc34999010d-1752834079.us-east-1.elb.amazonaws.com   8080:30954/TCP   33s
[root@eks ~]#
```

Both files are applied and pods, nodes are in running state. Using svc external ip is taken



Finally maven project is deployed in live . External_ip:8080/webapp to access the webpage.