Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To develop a website and host it on

a)Your Local Machine/VM

b)Amazon S3

c)Netlify

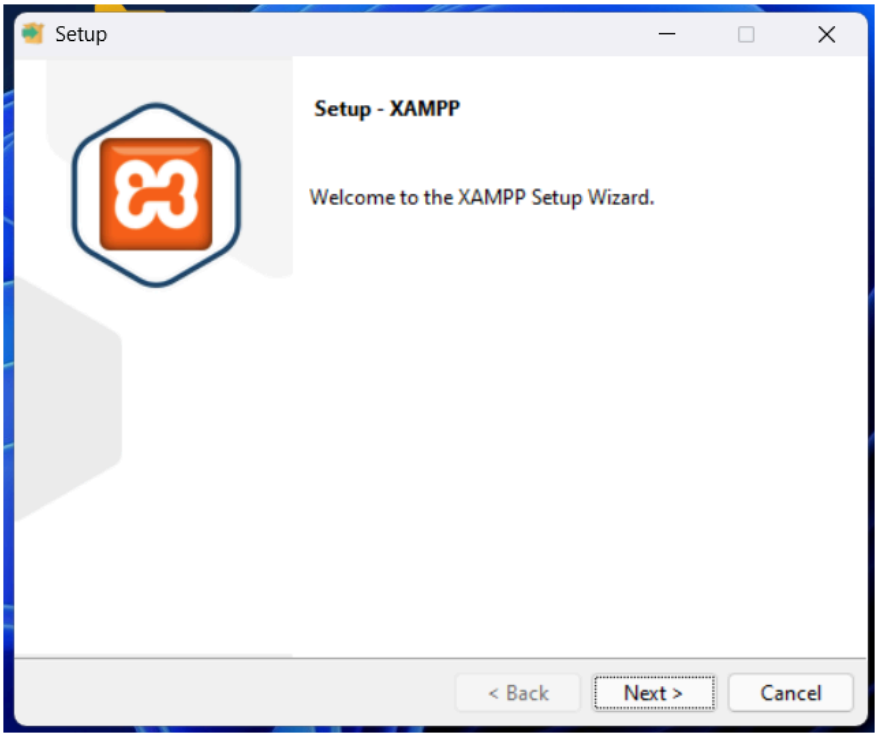
A)Static Hosting on Local Machine Using XAMPP:

XAMPP :

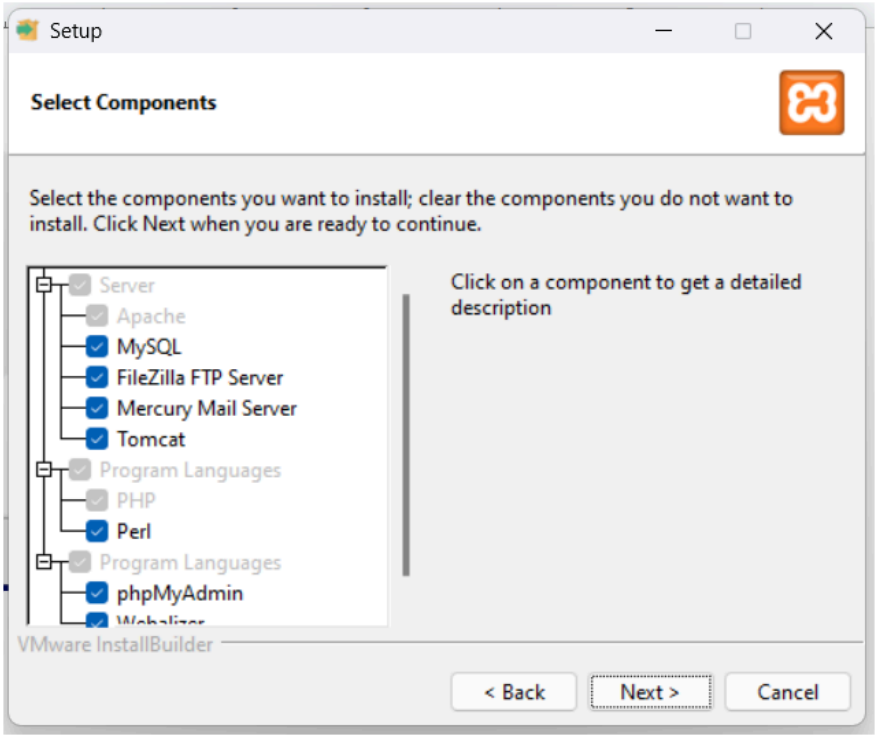
XAMPP is an open-source web server solution stack that provides a local server environment to run and test PHP, MySQL, and Perl-based web applications. It includes Apache, MySQL (MariaDB), PHP, and Perl, packaged together for easy installation. XAMPP is widely used for web development and testing before deploying websites live.

Step 1: Go to the XAMPP Website and Download it according to your Operating System. Here We have downloaded it for Windows.

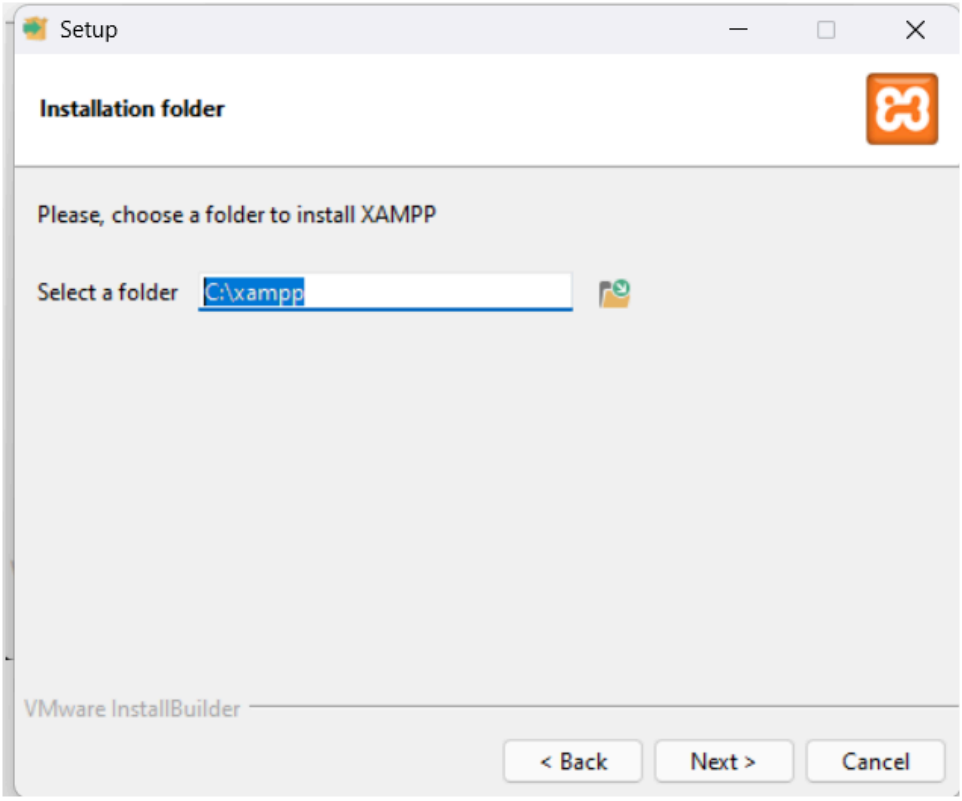
Step 2: Run .exe file.



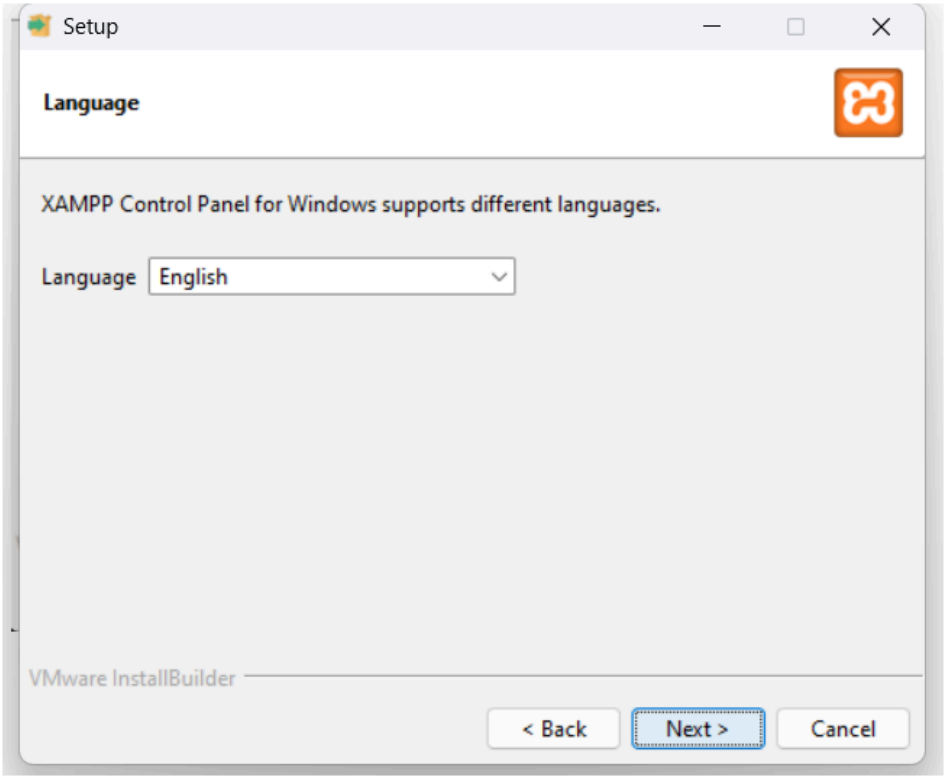
Step 3: Select Components as per your need Here we have selected all the components.

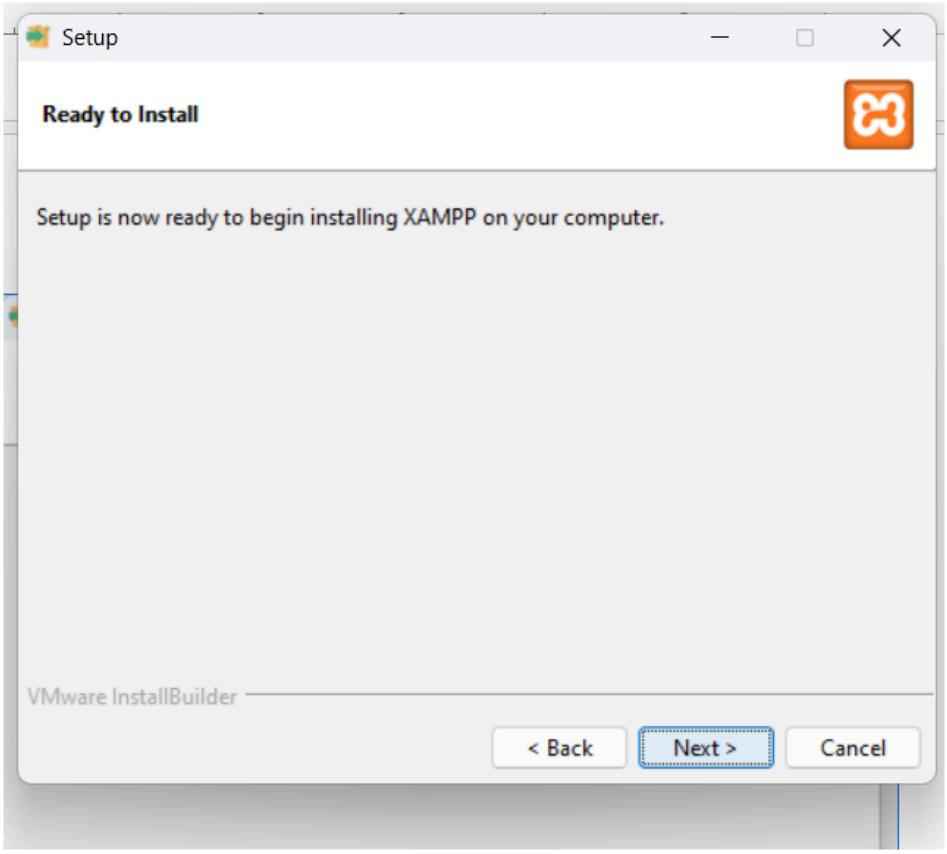


Step 4: Set path to C:\xampp

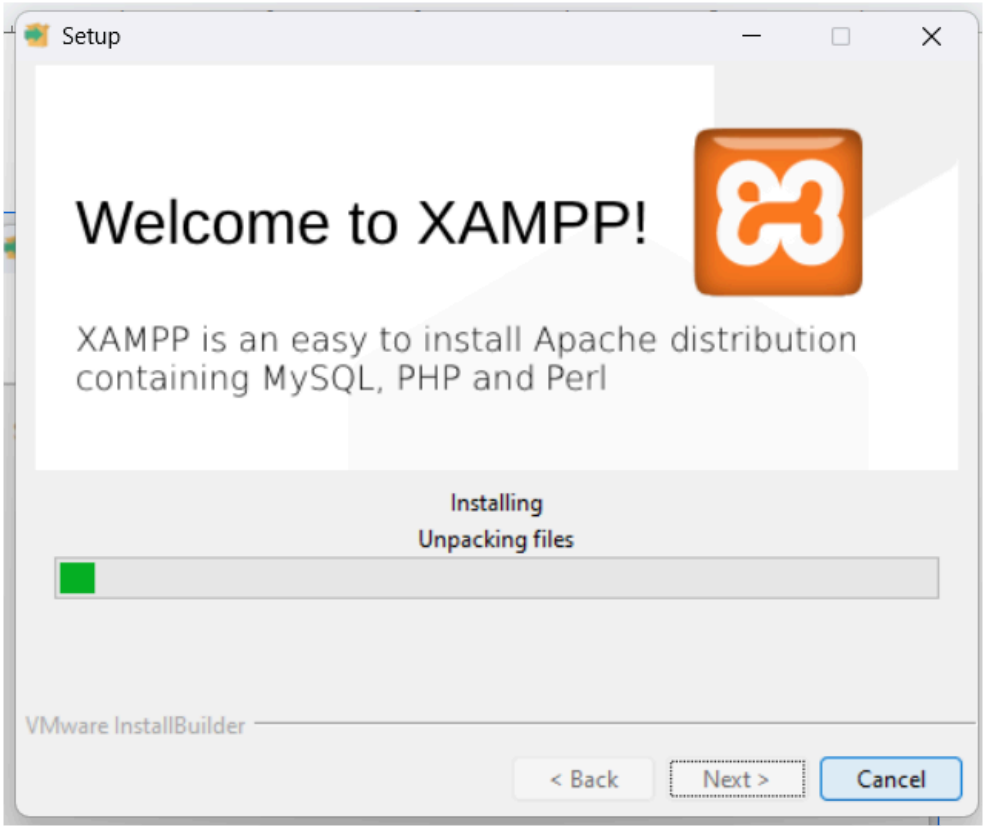


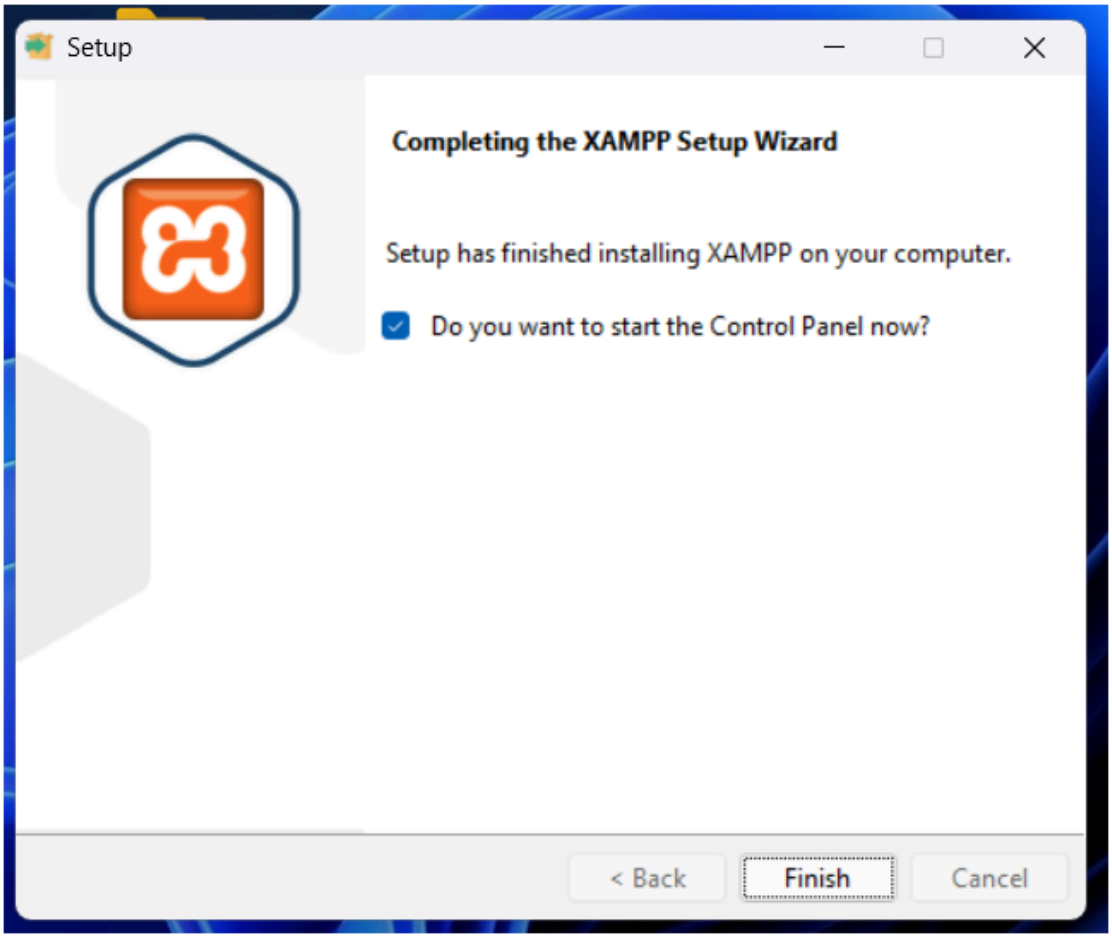
Step 5: Select language and Steup is ready to install click next.



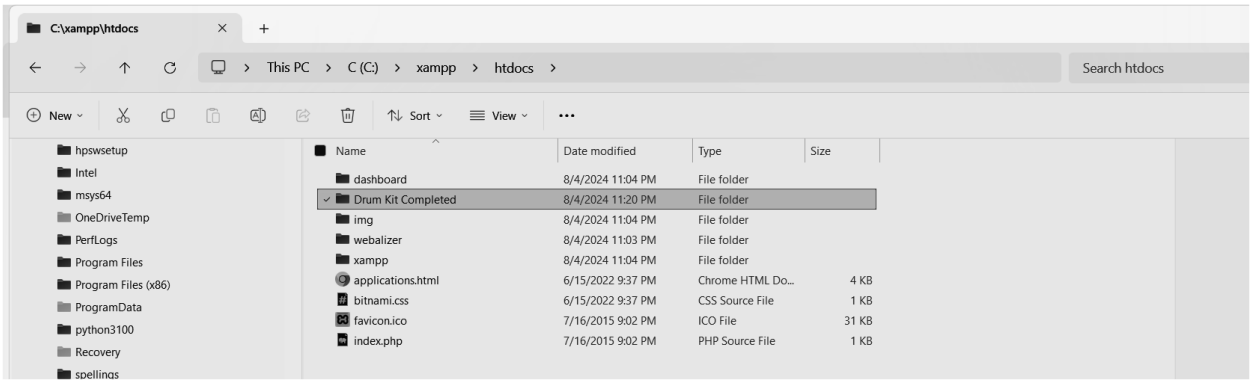


Step 6: Wait till completion of Installation then click on finish and Launch the XAMPP Control Panel.





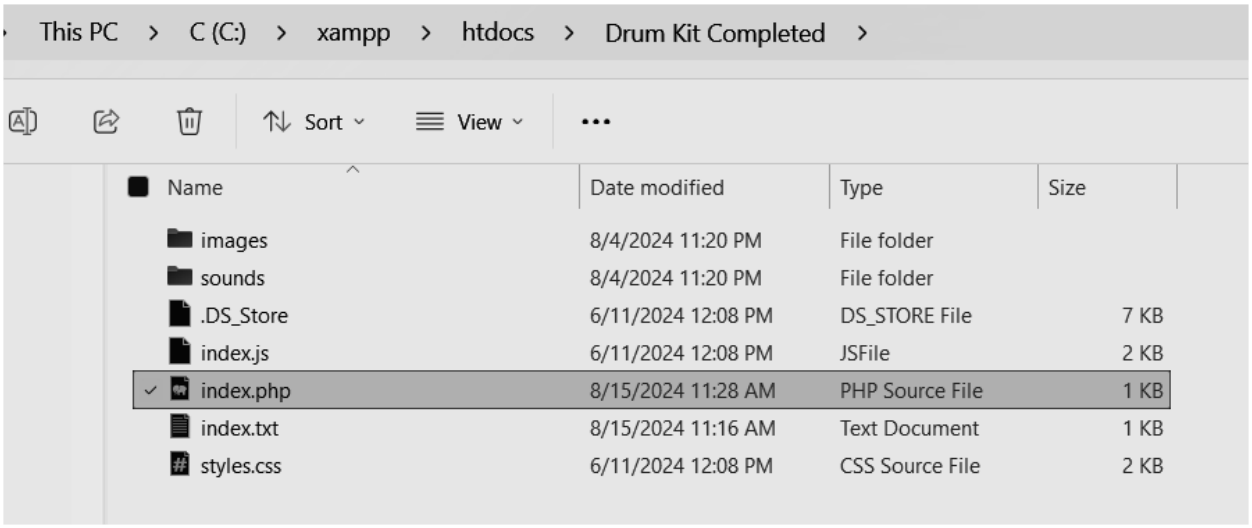
Step 7: Add your files in htdocs which present in location: **C:\xampp\htdocs**



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 8: If you have index.html change it into index.php and if you have to add some php feature then you can also add it in index.php file by adding php tag.




```
<!DOCTYPE html>
<html lang="en" dir="ltr">

<head>
  <meta charset="utf-8">
  <title>Drum Kit</title>
  <link rel="stylesheet" href="styles.css">
  <link href="https://fonts.googleapis.com/css?family=Arvo" rel="stylesheet">
</head>

<body>
  <h1 id="title">Drum 👹 Kit</h1>
  <div class="set">
    <button class="w drum">w</button>
    <button class="a drum">a</button>
    <button class="s drum">s</button>
    <button class="d drum">d</button>
    <button class="j drum">j</button>
    <button class="k drum">k</button>
    <button class="l drum">l</button>
  </div>


<script src="index.js" charset="utf-8"></script>

<footer>
  Made with 🖤 in India.
</footer>
</body>
</html>
```

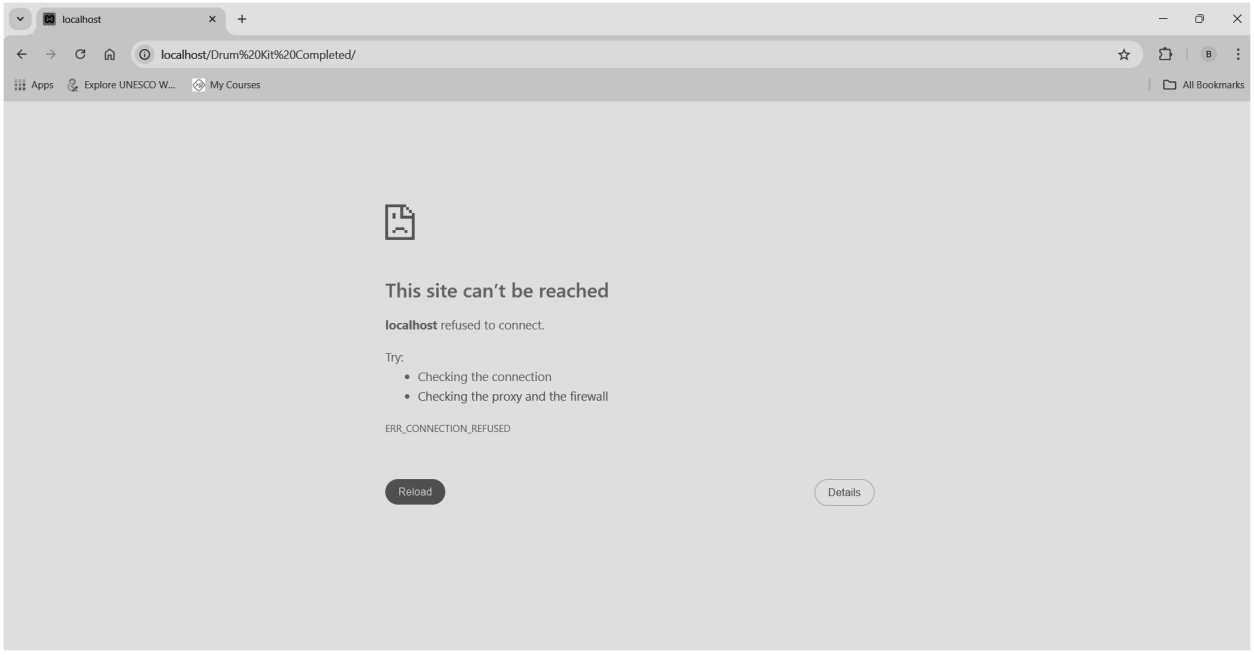
```
<?php
  // Sample PHP code
  $title = "This Line is Coming from Echo in PHP part.";
  echo "<h2 style='color:white' id='title'>$title</h1>";
?>
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

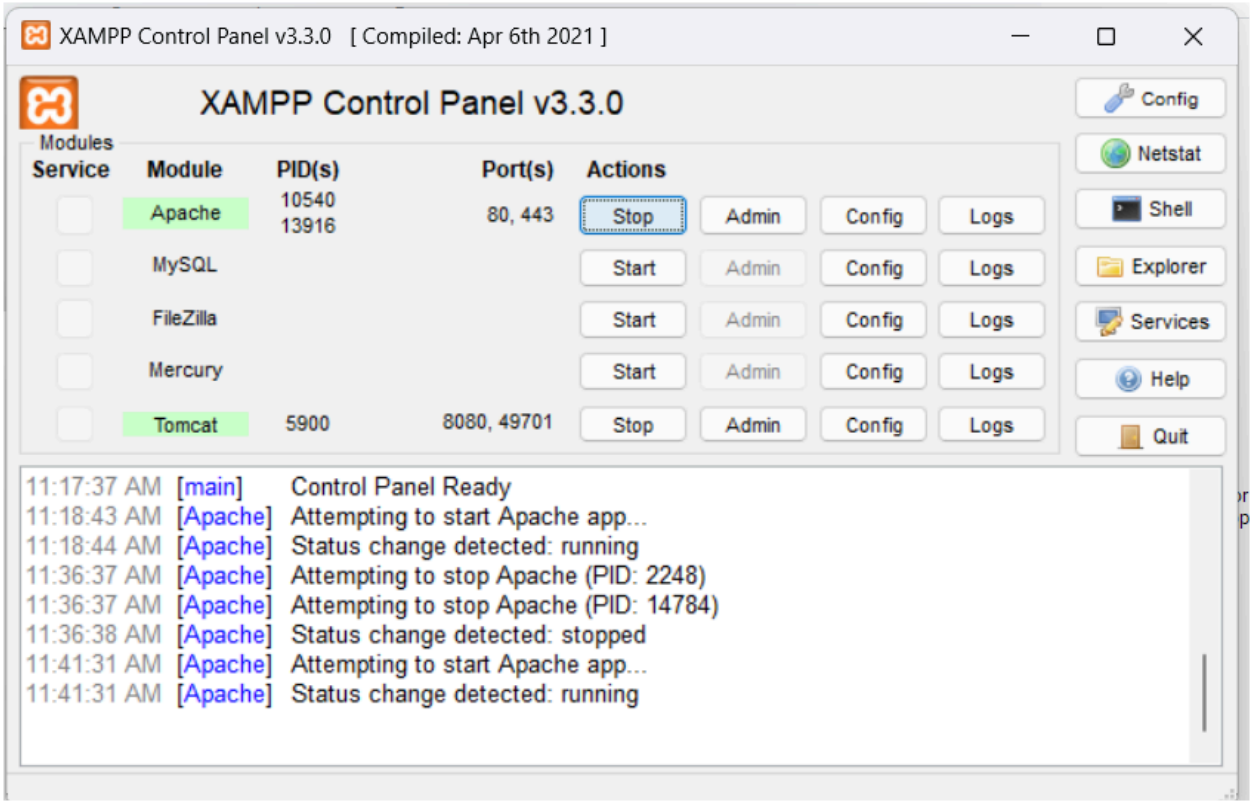
Division: D15C **Roll No: 28**

Step 9: Start the Server and type “http://localhost/Drum Kit Completed/” in the browser or localhost/your filename

Before Starting Server



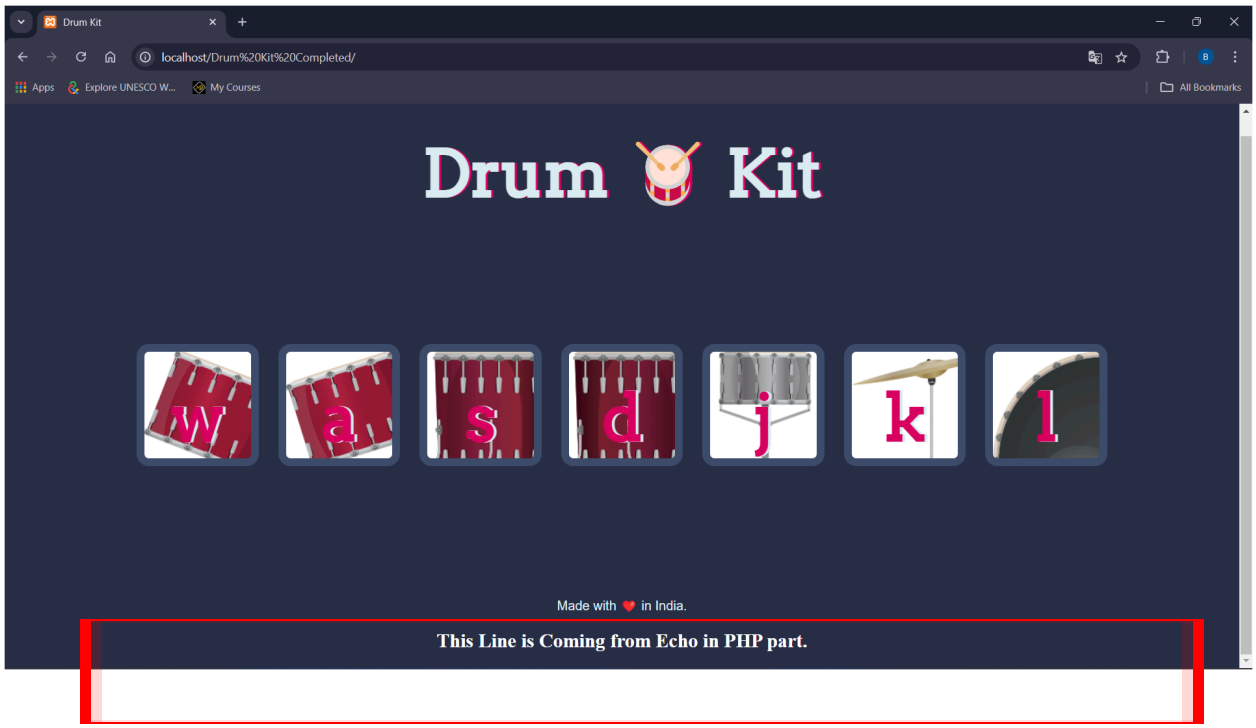
Start the Server



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

After Starting the server refresh it we can see html and at the end php part



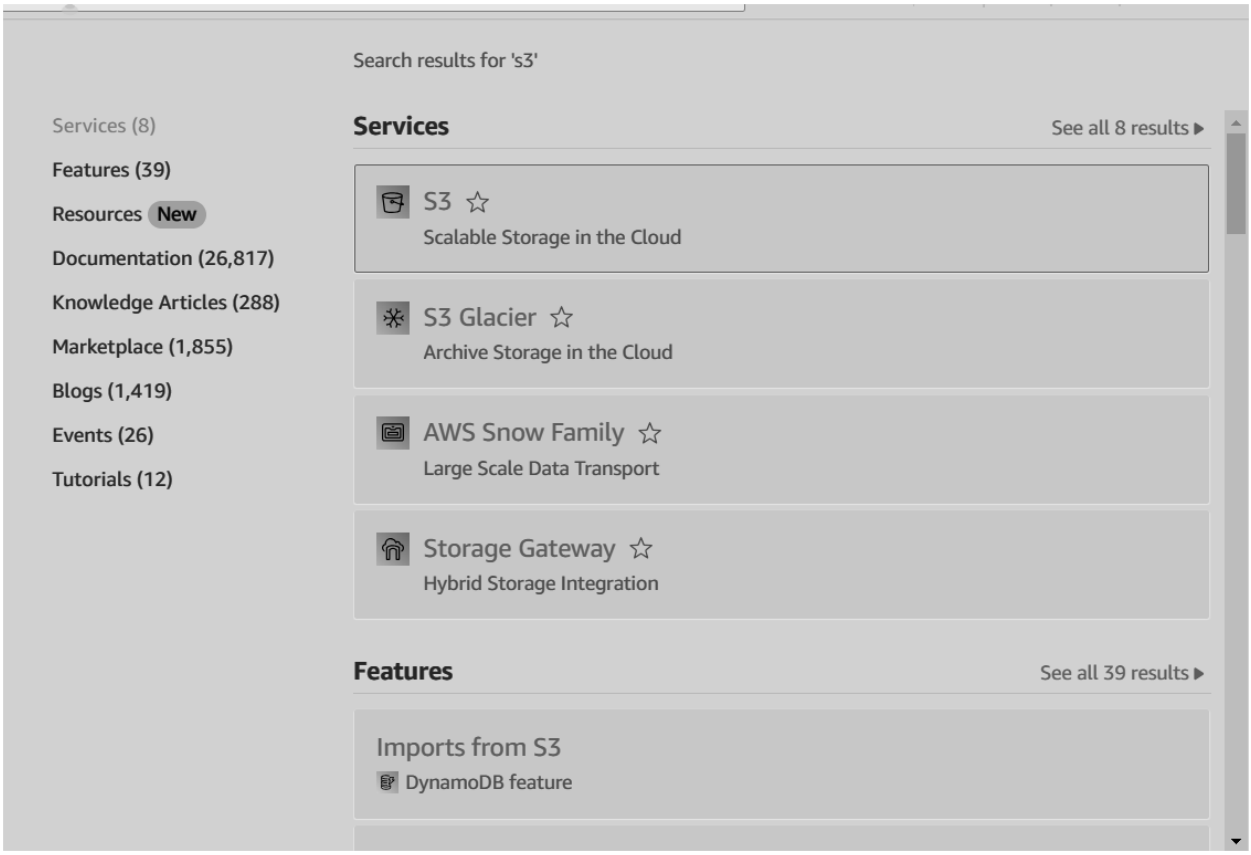
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

B)Stating hosting Using Amazon S3 Bucket

Amazon S3: Amazon S3 (Simple Storage Service) is a scalable cloud storage service provided by AWS, designed for storing and retrieving any amount of data at any time. It offers high availability, durability, and security, making it ideal for a wide range of use cases such as backup, archiving, and big data analytics. S3 also supports a pay-as-you-go pricing model, allowing businesses to optimize costs.

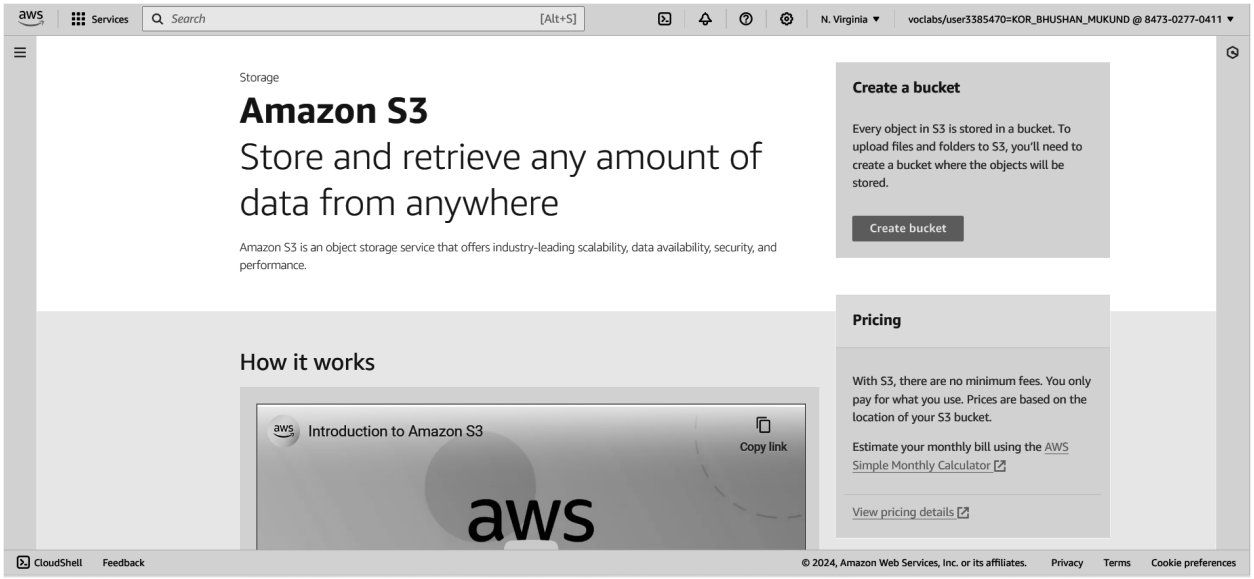
Step 1: Open Your AWS academy account or Personal AWS account .Here we have used AWS academy account. Then Search for S3 in Servies and click on it.



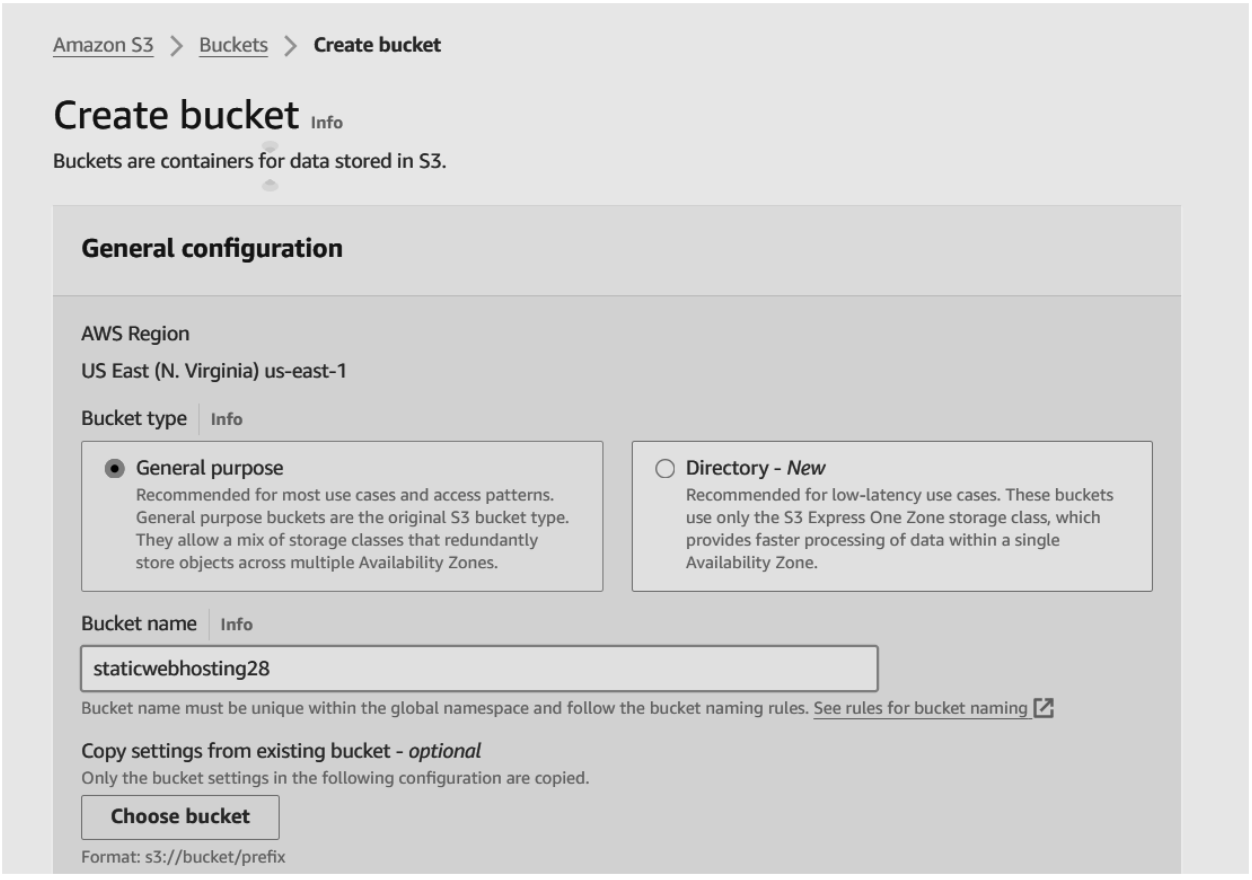
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

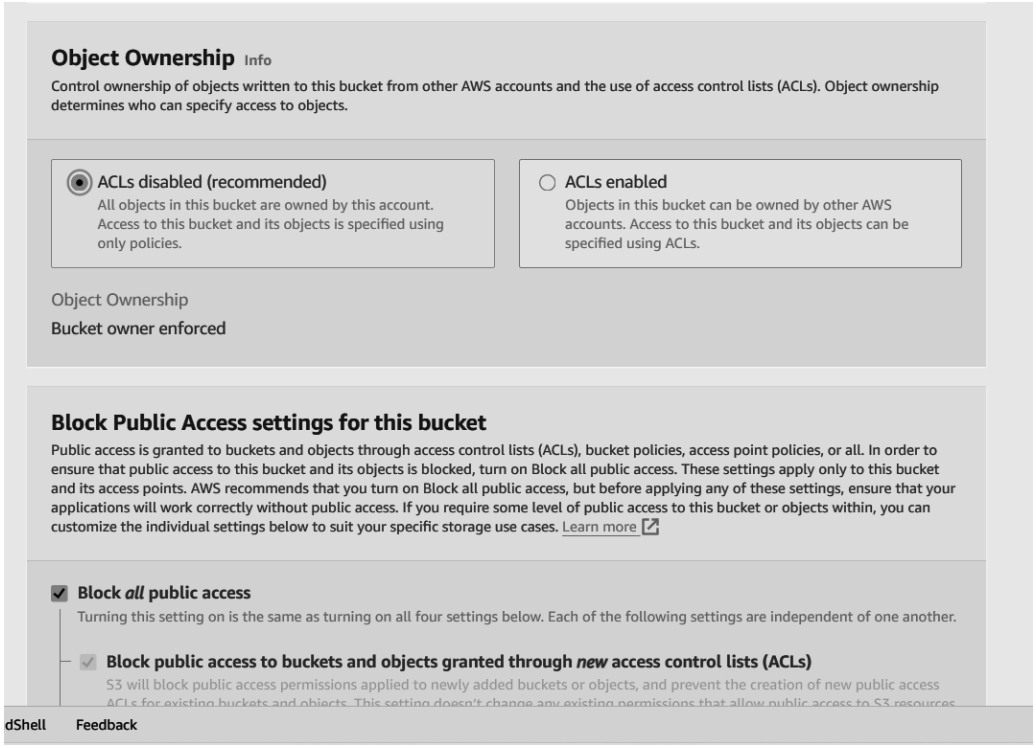
Step 2: Click on Create Bucket Button.

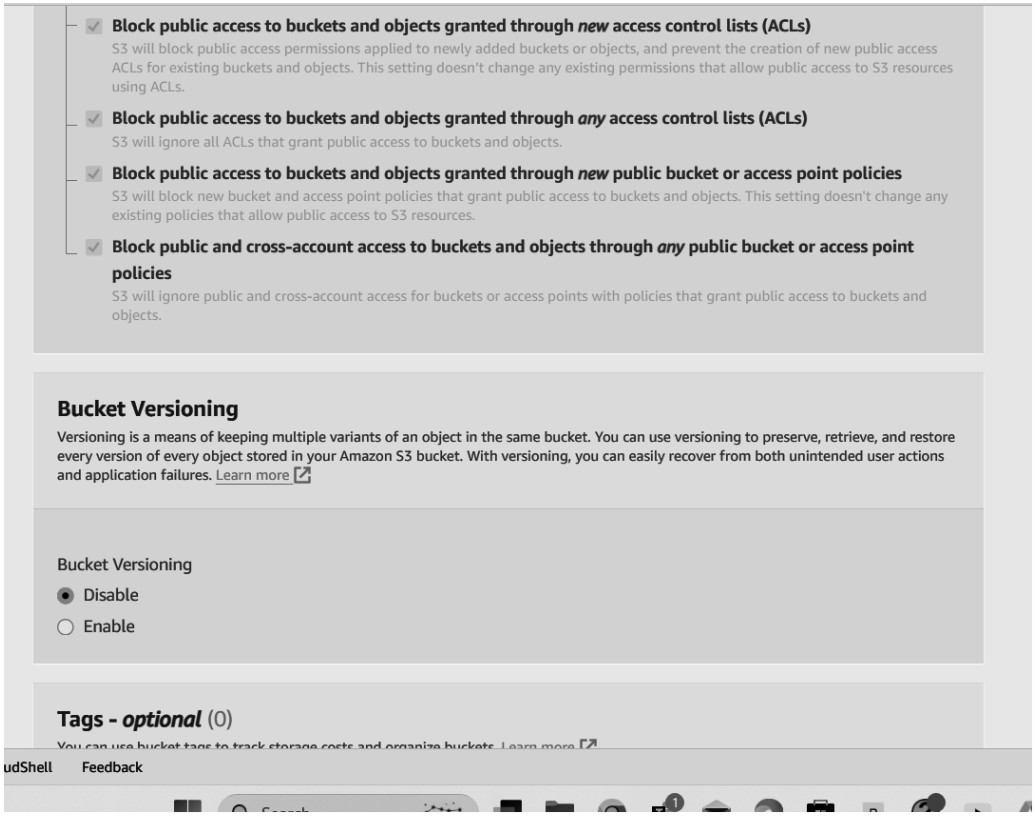


Step 3: Select Bucket type: General Purpose then give Bucket name.

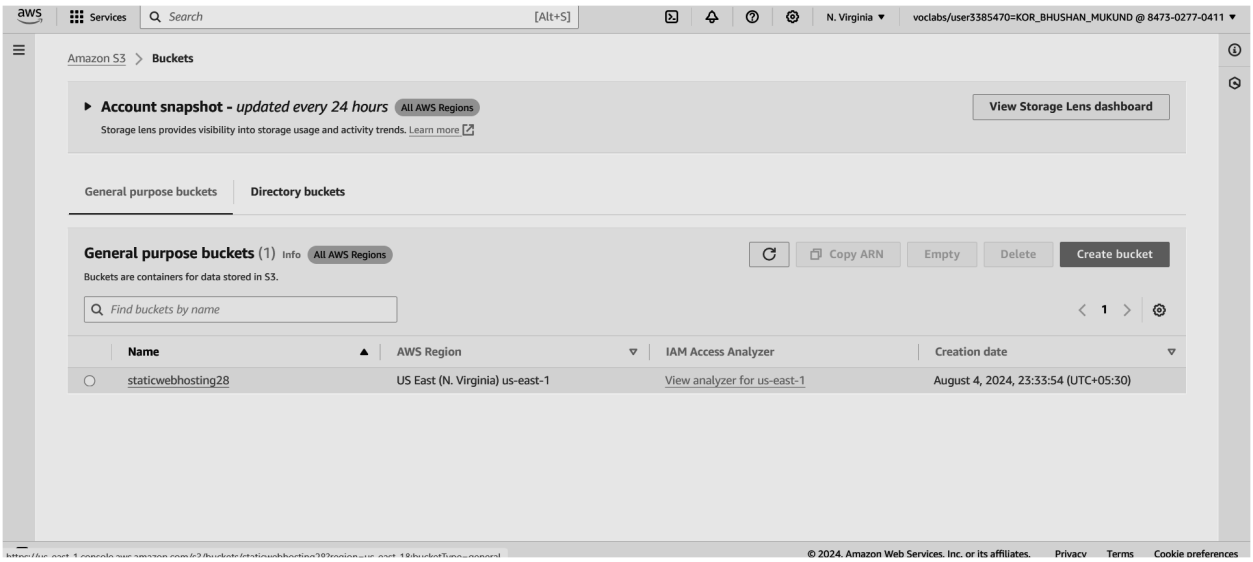


Step 4: Keep all the other things By Default Refer to screenshots. And Click on Create Bucket.



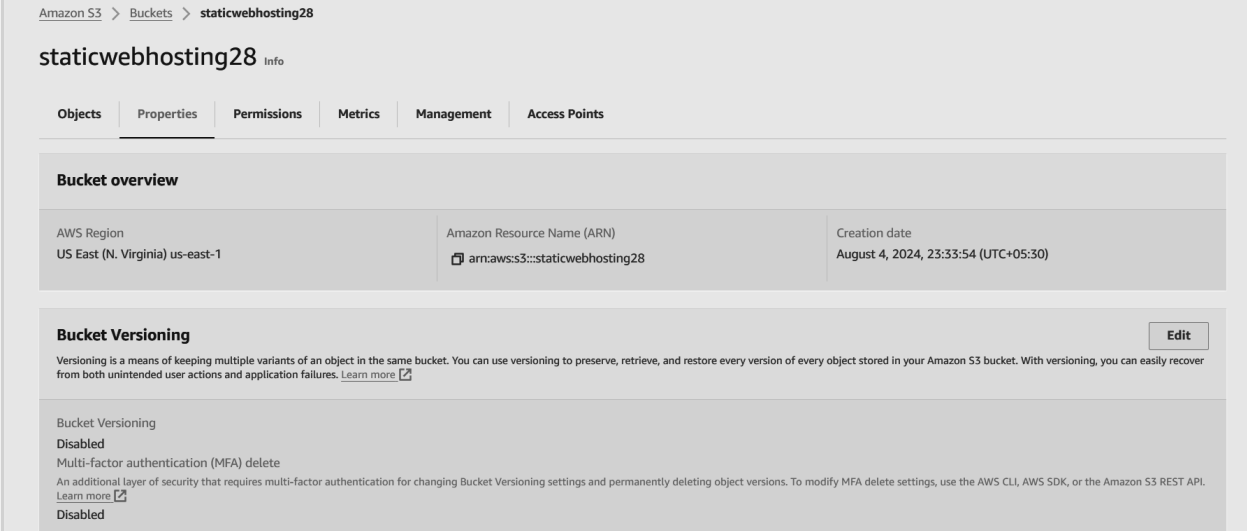


Step 5: Bucket is created Successfully.

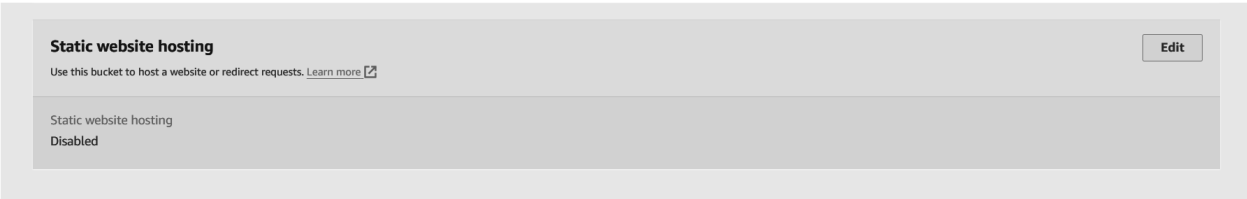


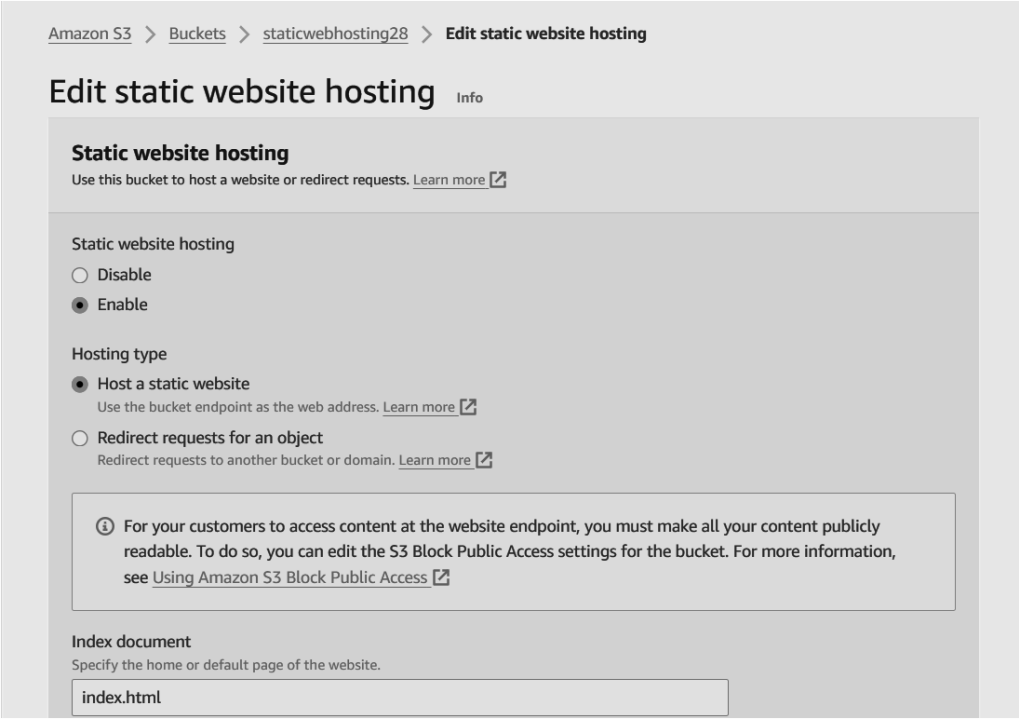
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



Step 6: By Default Static hosting is disabled so enable it.Also give name of index document here it is index.html

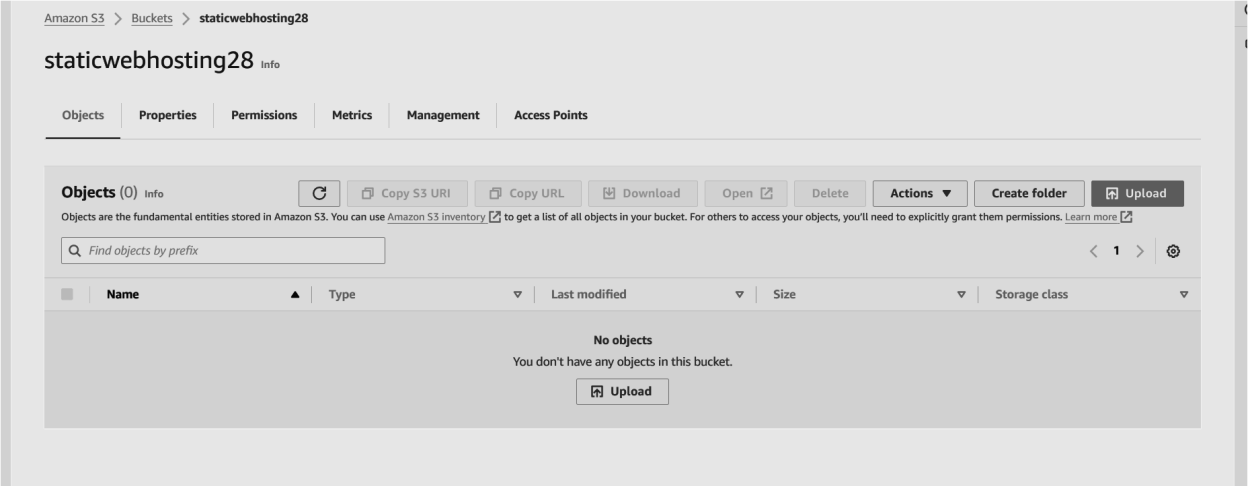


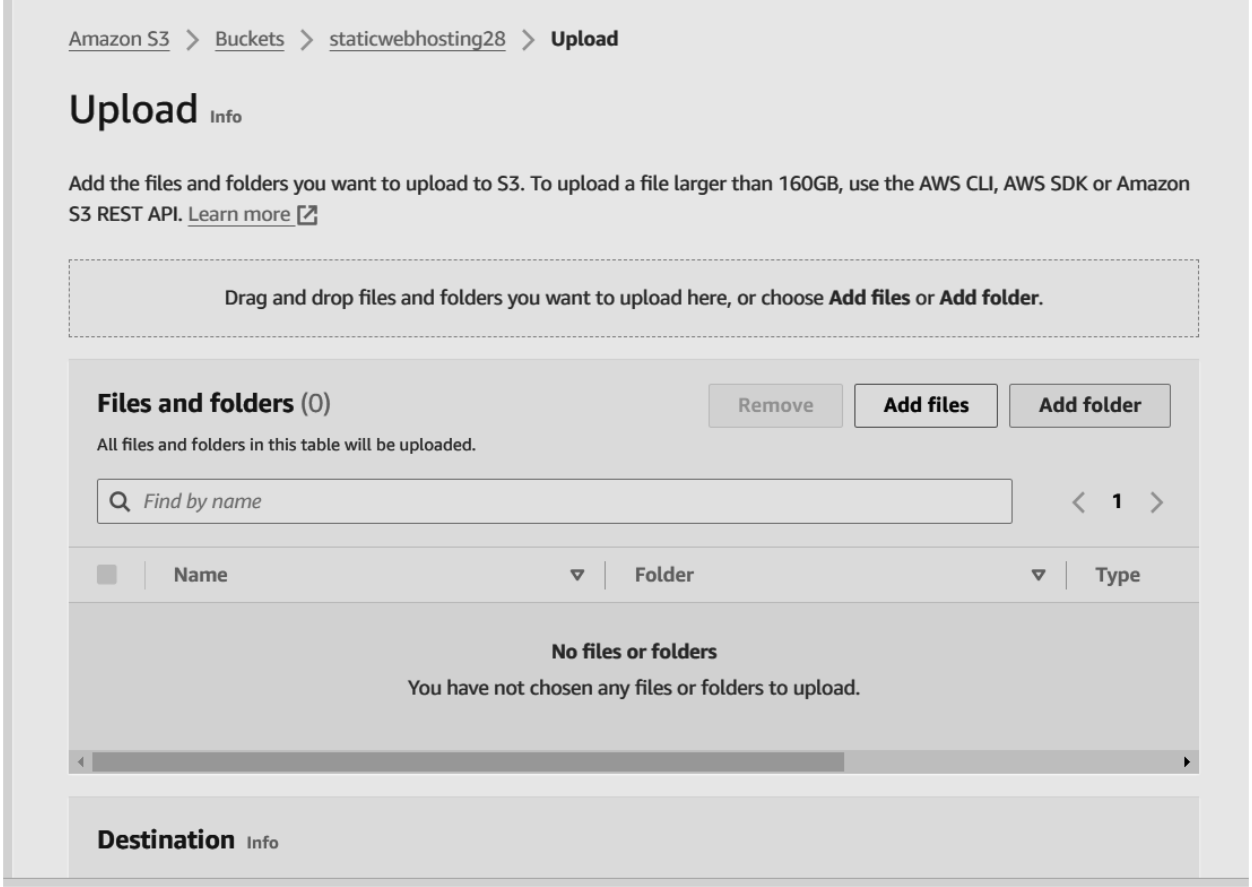


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

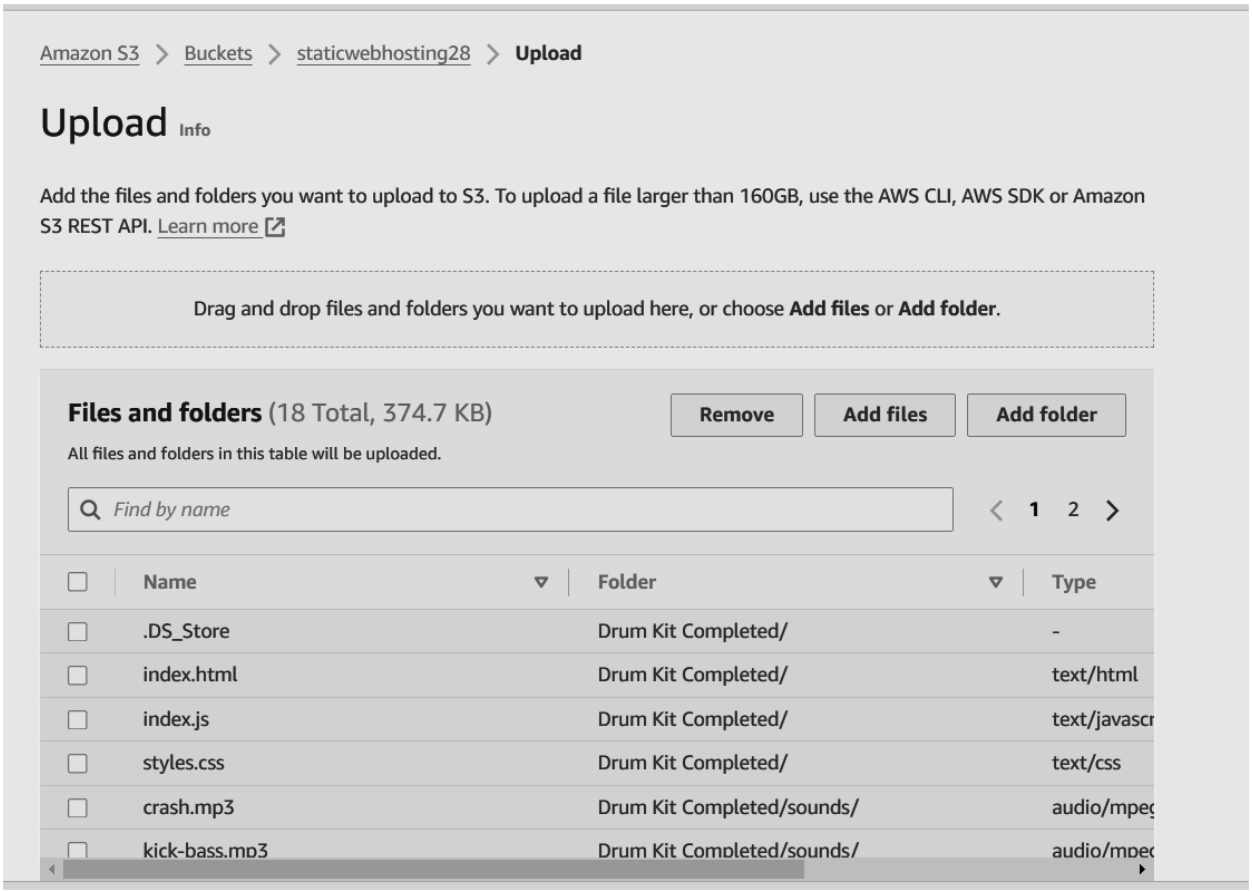
Step 7: Click on the Bucket name and then Upload your files in Bucket by clicking on the upload button refer screenshots.

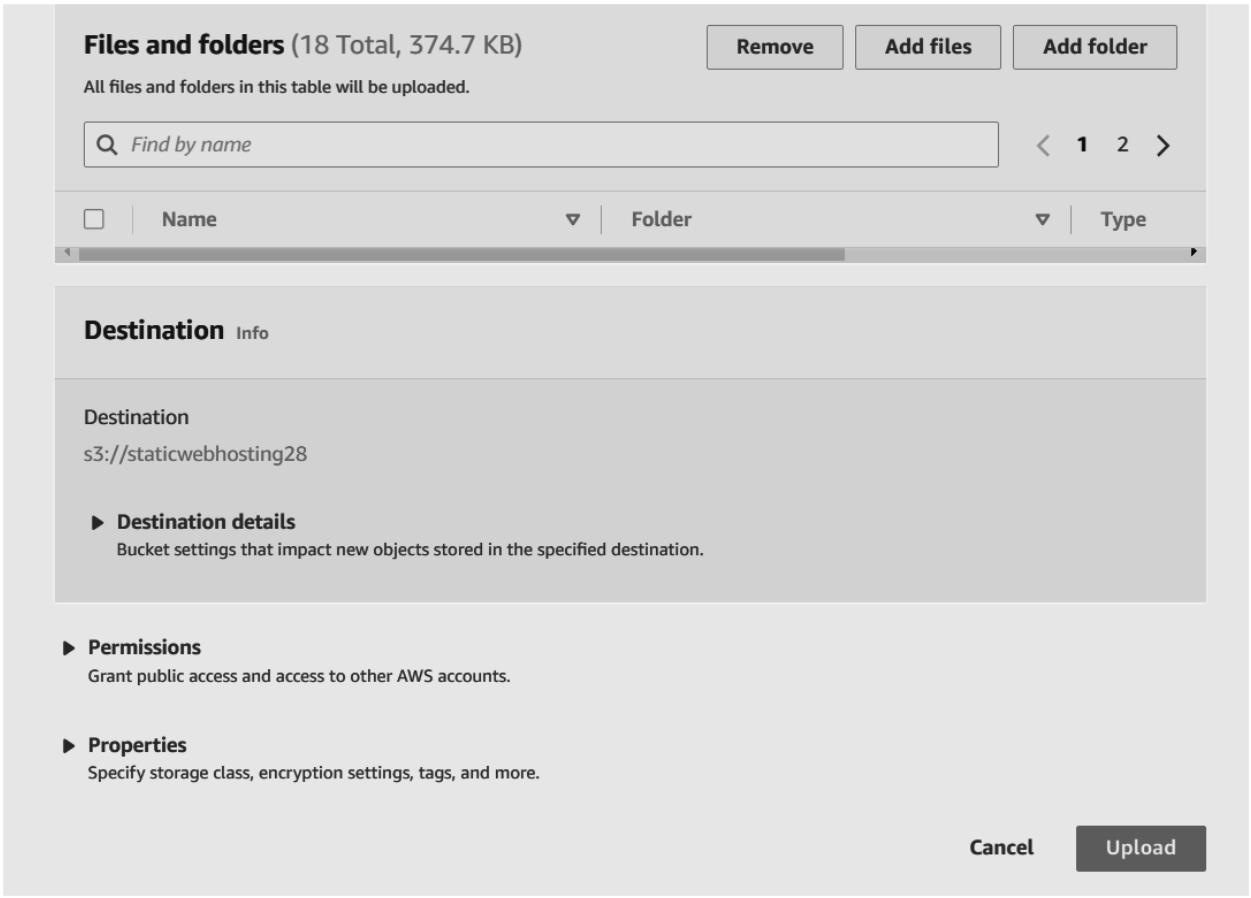




Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**





Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

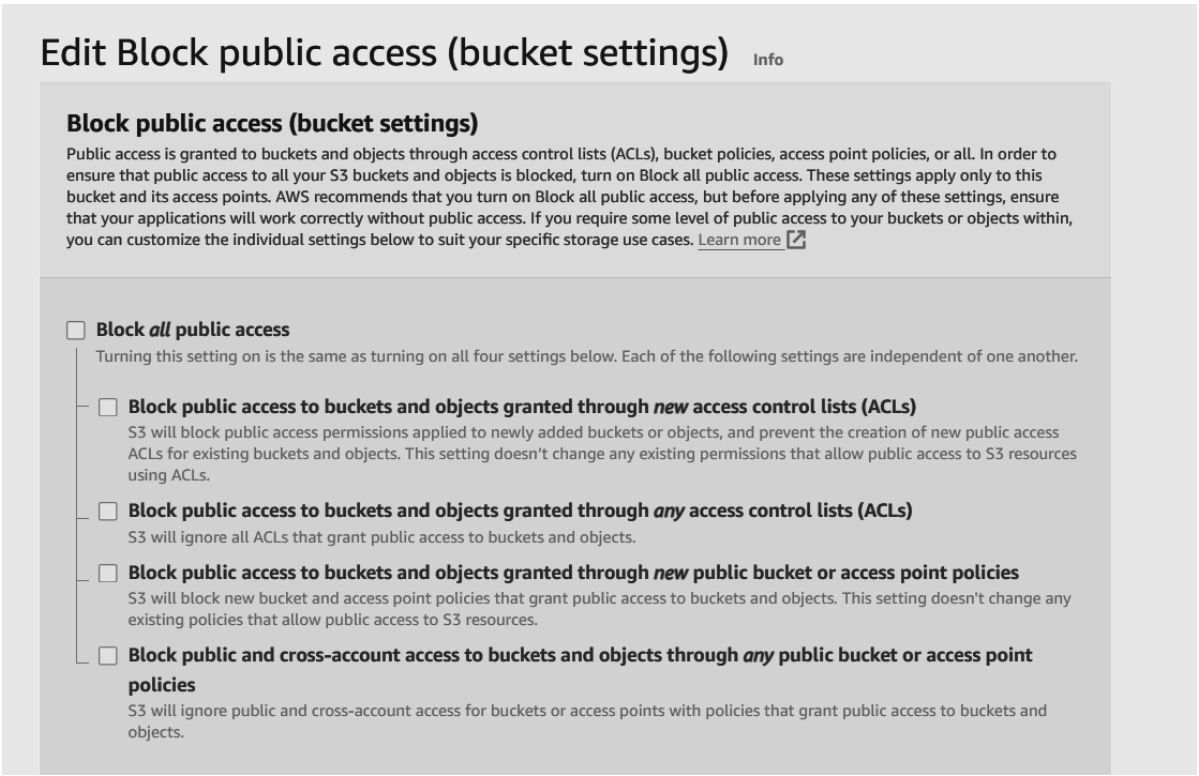
Step 8: Now click on the link.Then you will get to know access is block so change the access settings.

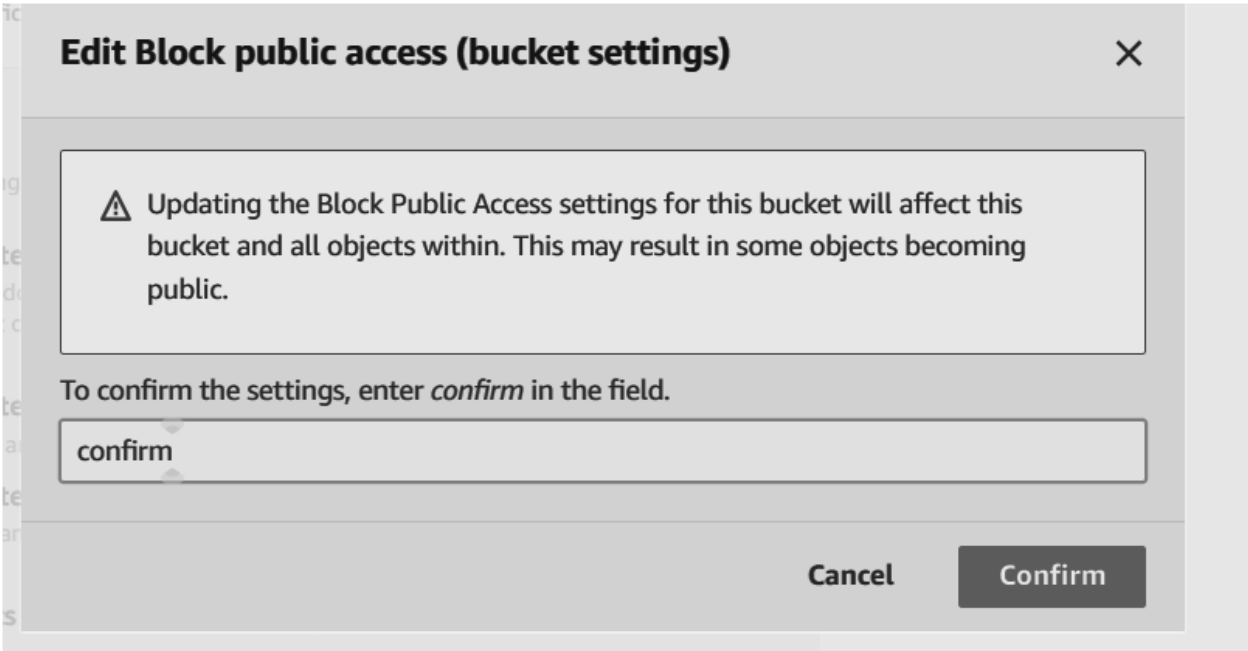
403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: BG0328HP96F83TQ1
- HostId: OfGAgKa7cFKwwCAVutc37YPwLicC9TNEfOWsgROBYudamAXB79gJbcBFAMbT16vPX6bO8A49wcE=

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
- Message: Access Denied



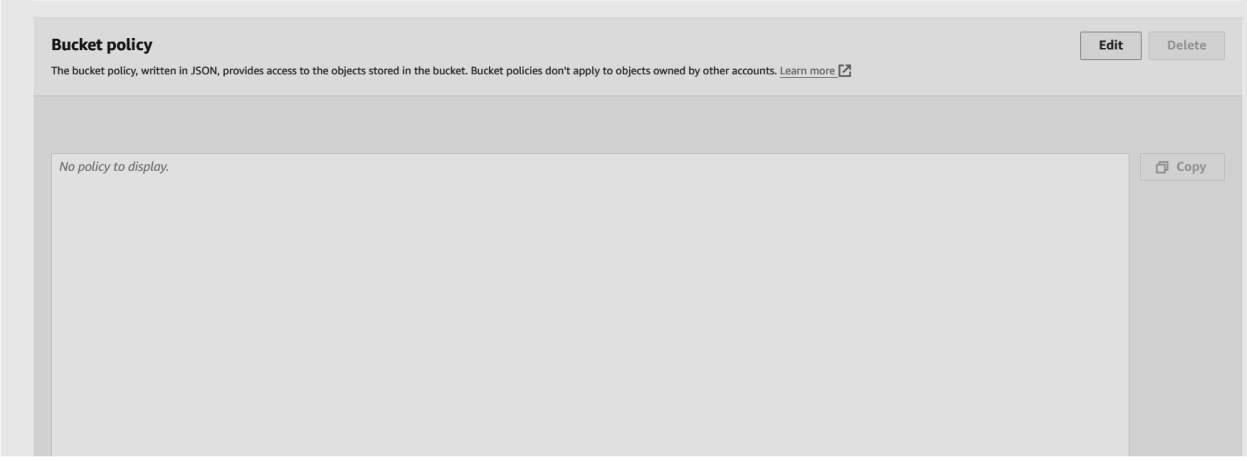


Step 9: The website is still not working because there is one more step that you have to add some policies so click on edit policy. And Paste the below policy in the box and save it.
Do not forget to replace your Bucket Name.

```
{
 "Version": "2012-10-17",
"Statement" : [
{
"Sid": "PublicReadGetObject",
"Effect ": "Allow",
"Principal": {
"AWS": "*"
},
"Action": "s3:GetObject",
"Resource": "arn:aws:s3: : :YOUR-BUCKET-NAME-HERE/*"

}
]
}
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

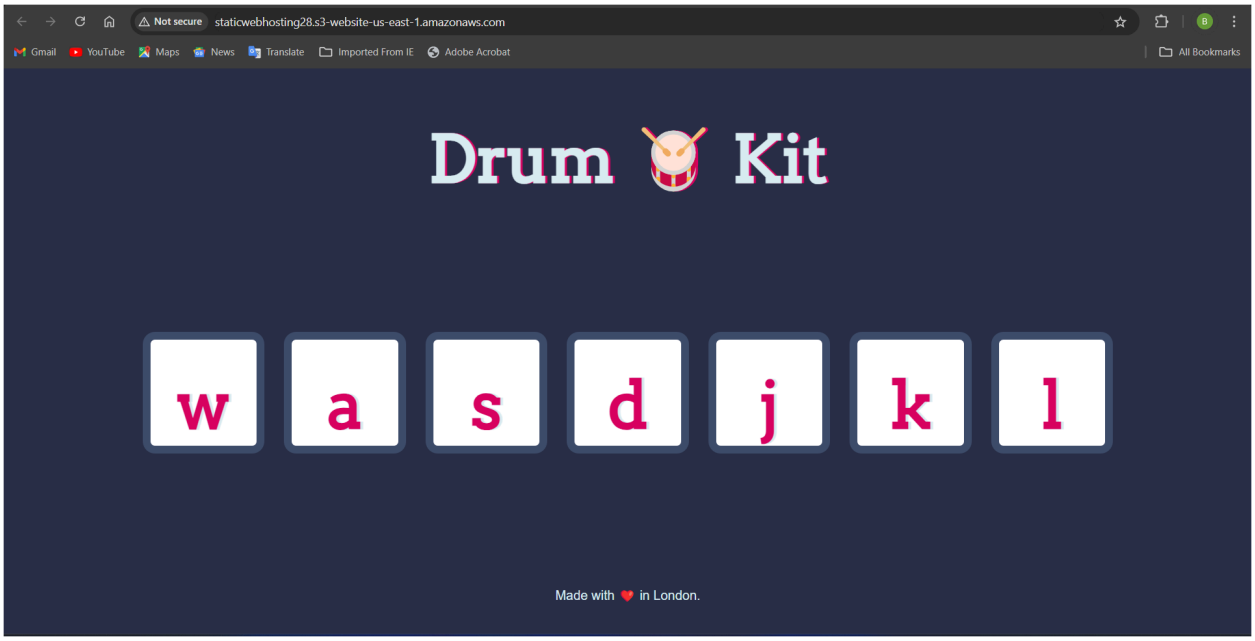
Division: D15C **Roll No: 28**



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 10: Done Now go to the link where you can see your website.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

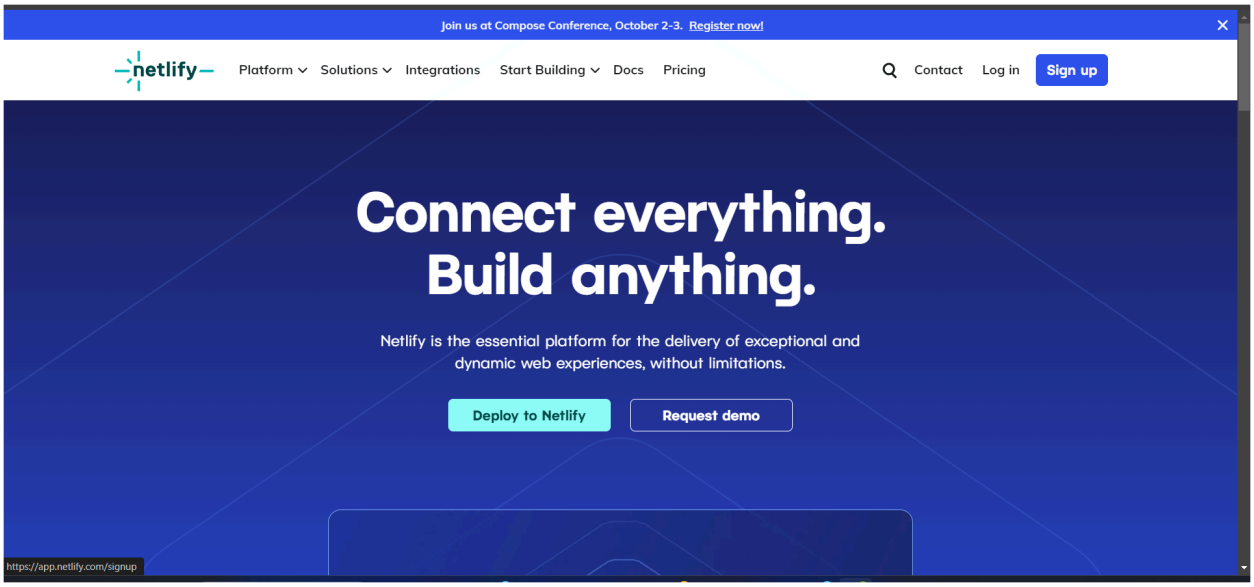
C)Static Hosting Using Netlify

Netlify

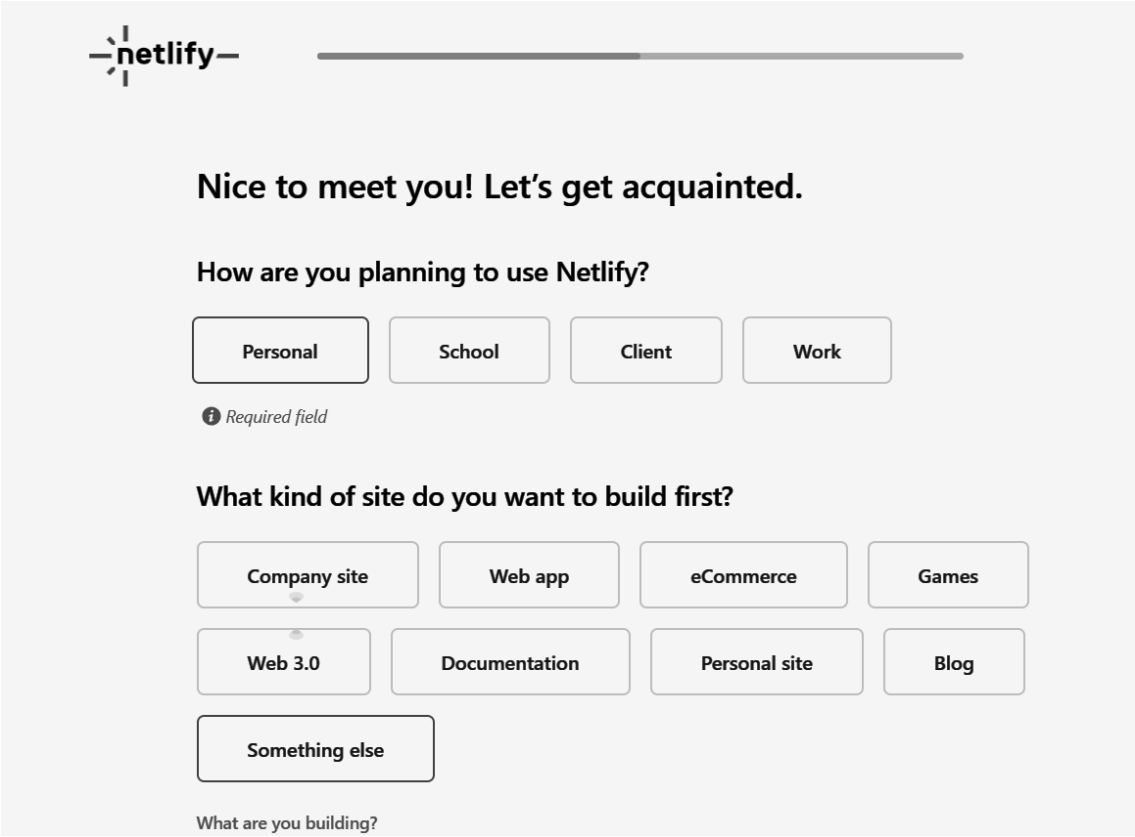
Netlify is a cloud platform that automates the deployment, management, and hosting of modern web projects. It provides developers with tools for continuous integration, automatic builds, and seamless deployment from Git repositories. Netlify supports static sites and serverless functions, making it ideal for JAMstack applications.

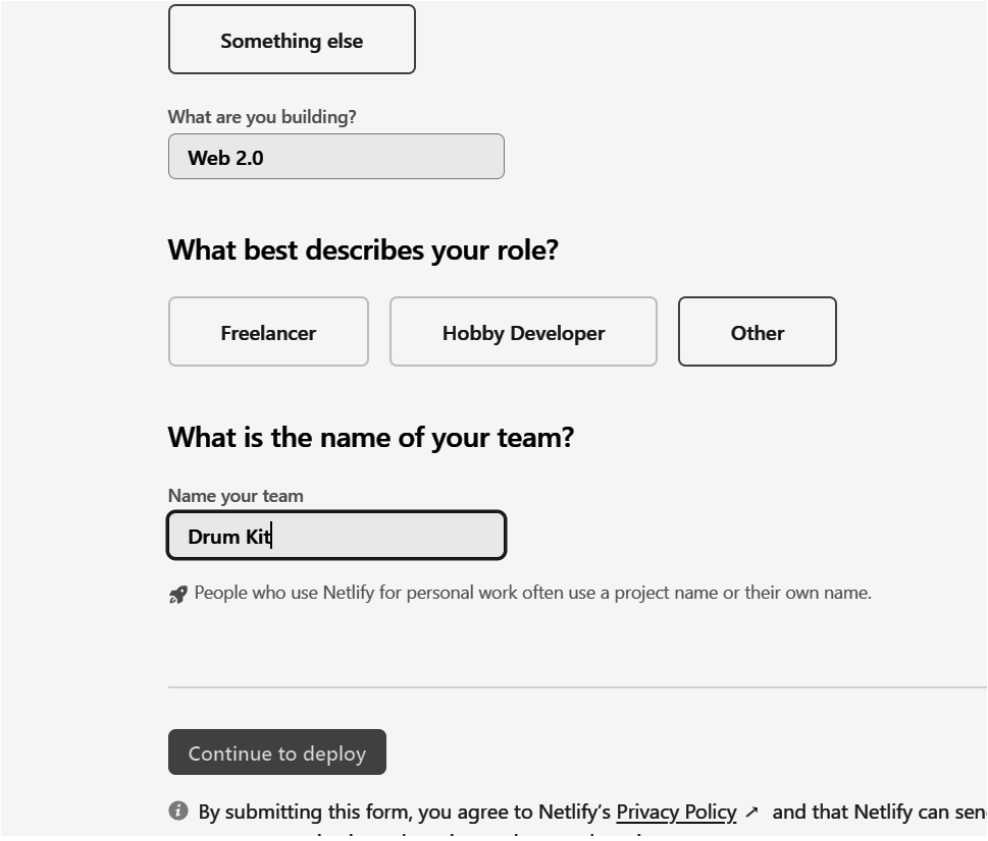
It is the easiest and fastest way to host a website.

Step1: Create your account or Sign up with a Google account on Netlify.



Step 2: Give Some Basic Information Regarding your web application. Add your files or connect to a GitHub account.

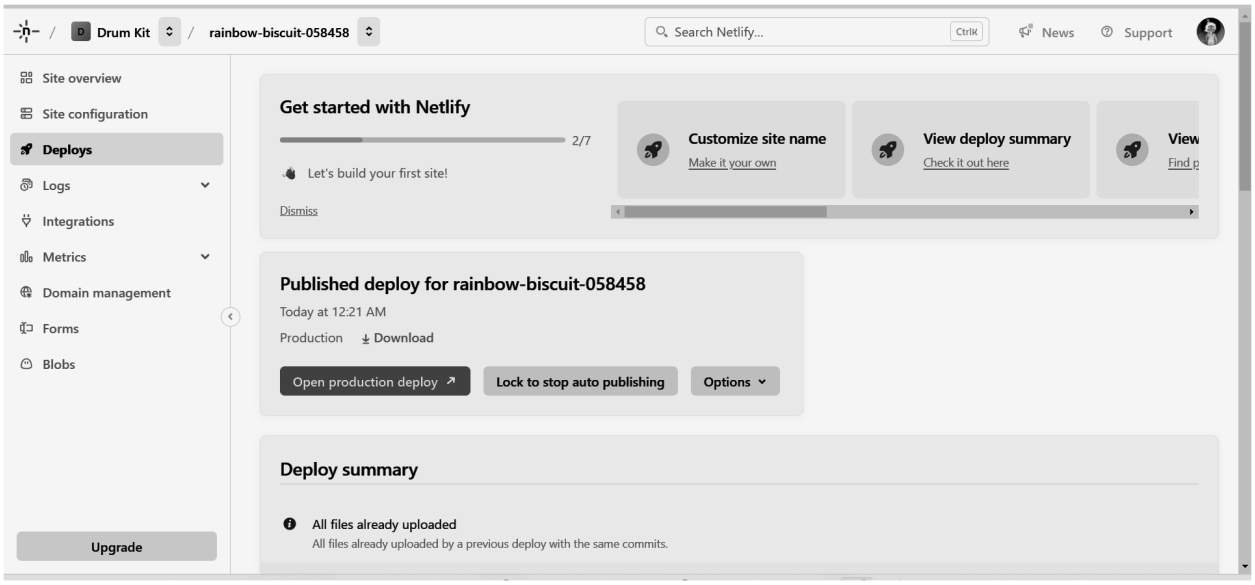


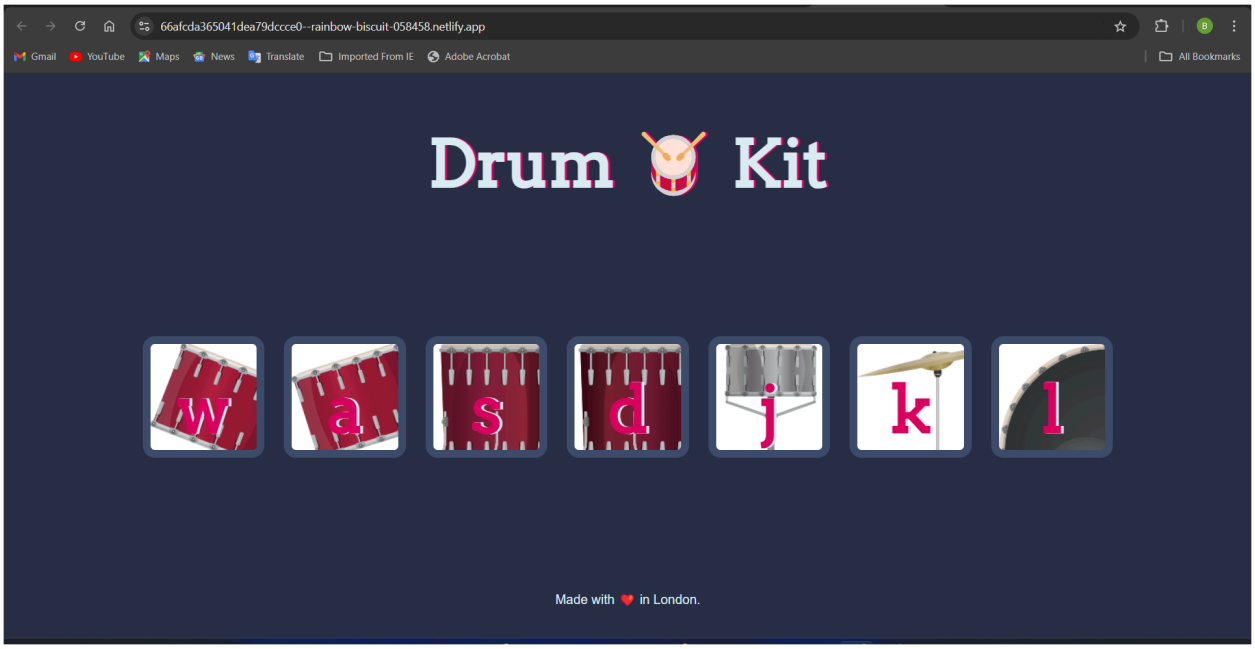


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 3: Click on the link where you can access your web application and you can also share this link with your friends.





Step 4: Done. This is the Public link.

Link: https://66afcda365041dea79dccce0--rainbow-biscuit-058458.netlify.app/

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

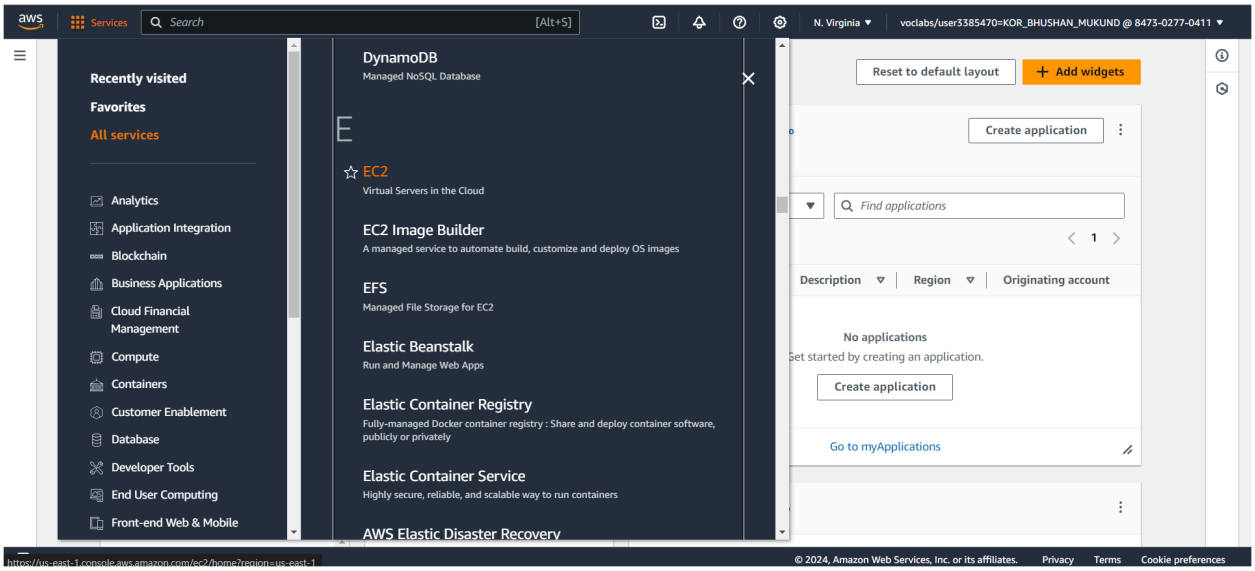
Aim: To understand the benefits of Cloud Infrastructure, Setup EC2 and Setup AWS Cloud9 IDE, Launch AWS Cloud9 IDE, and Perform Collaboration Demonstration.

A)EC2

EC2

Amazon EC2 (Elastic Compute Cloud) is a web service that provides resizable compute capacity in the cloud. It allows users to run virtual servers, known as instances, on demand. EC2 offers flexibility in scaling resources up or down based on your needs, making it ideal for hosting applications, running batch jobs, and more, with only the cost of the resources you actually use.

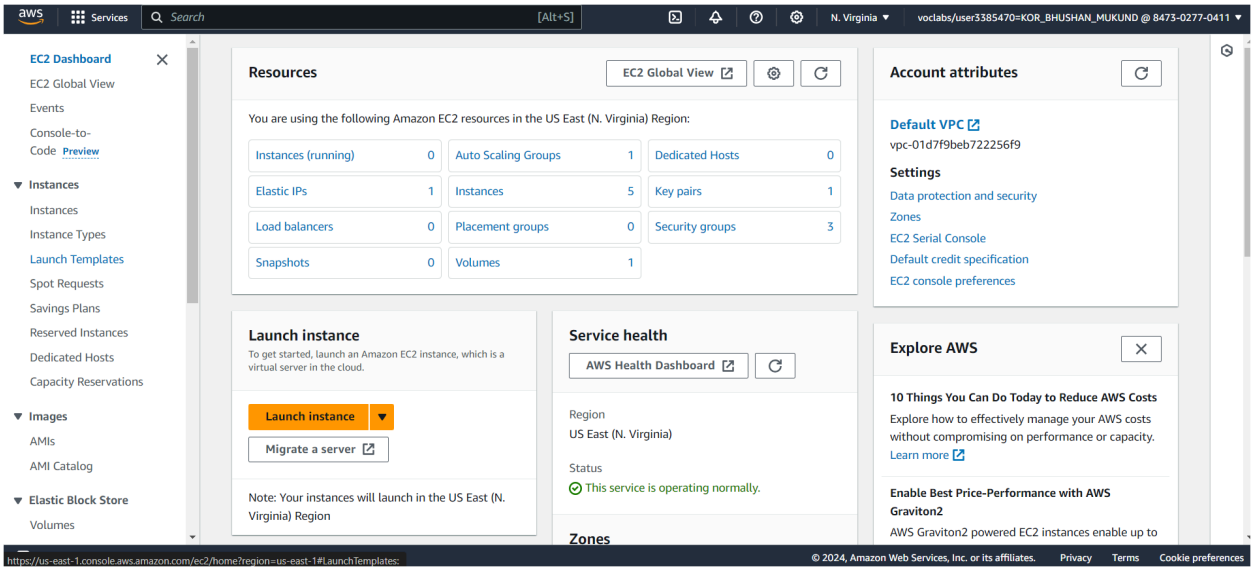
Step 1: Open Your AWS Academy or Personal AWS account and Search for EC2 in services.



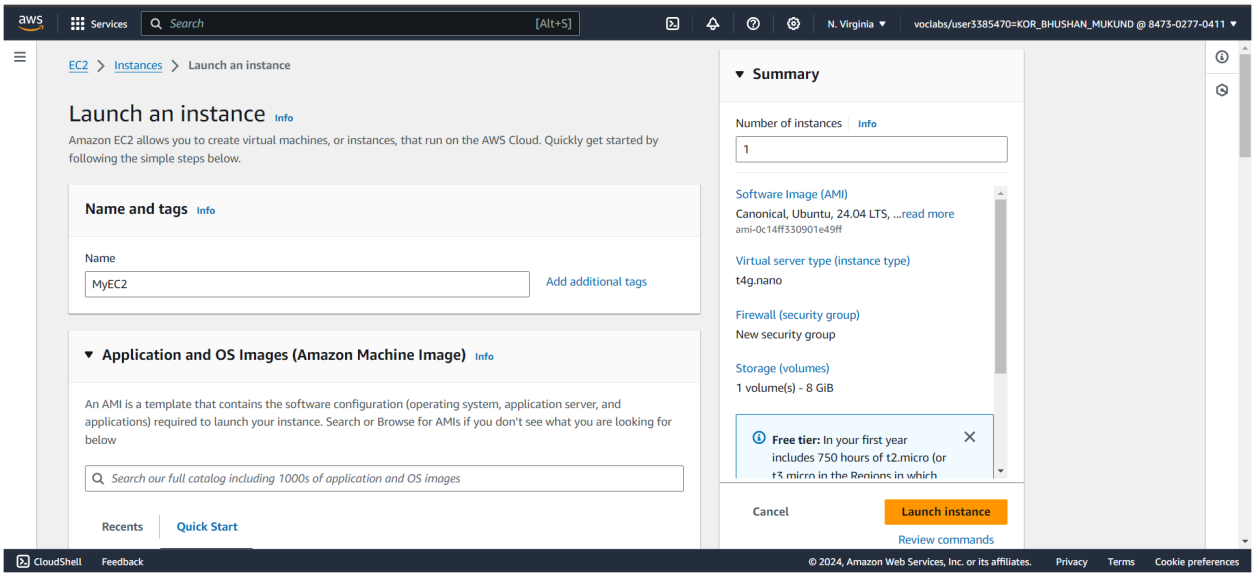
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 2: After Opening EC2 click on Launch instance.



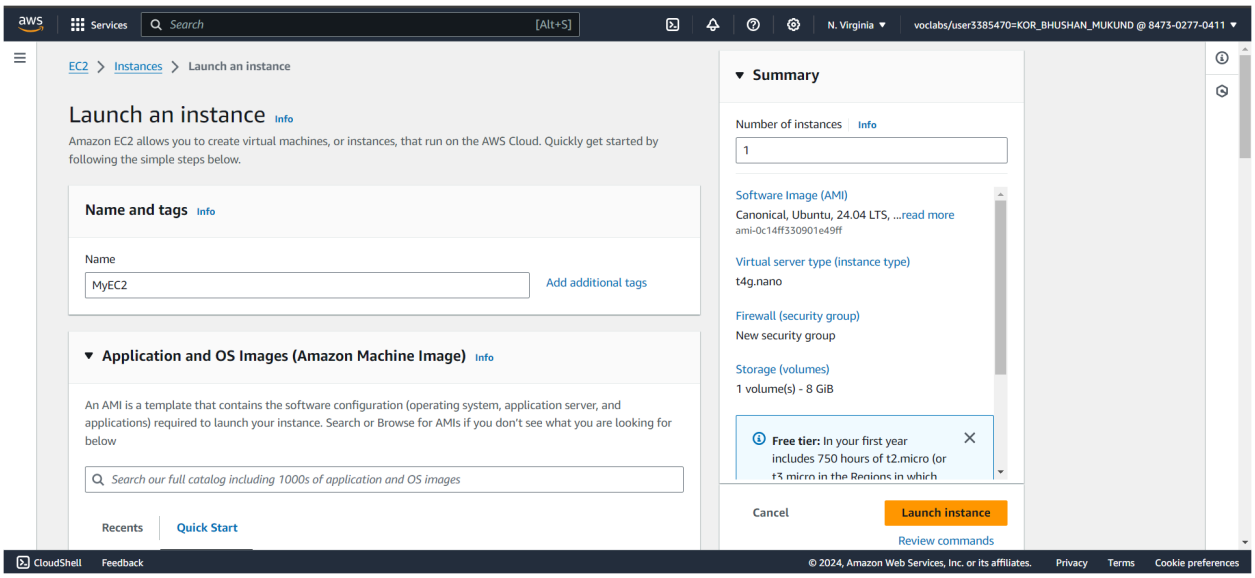
Step 3: Give a name to your instance.



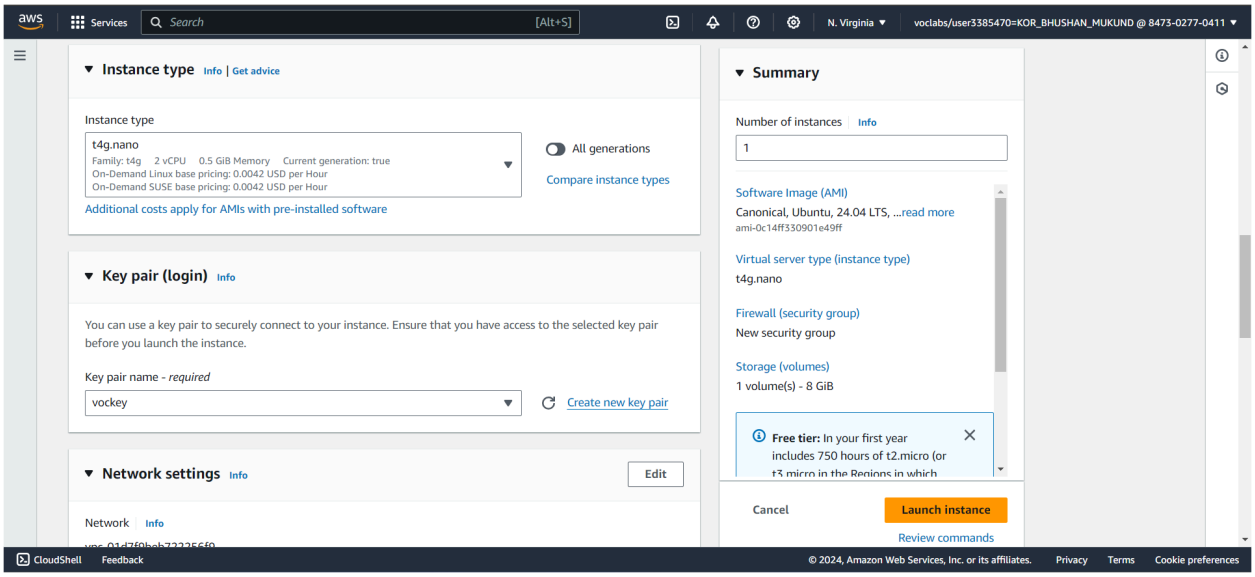
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 4: Select the server as Ubuntu and you can select Architecture x86 or ARM .



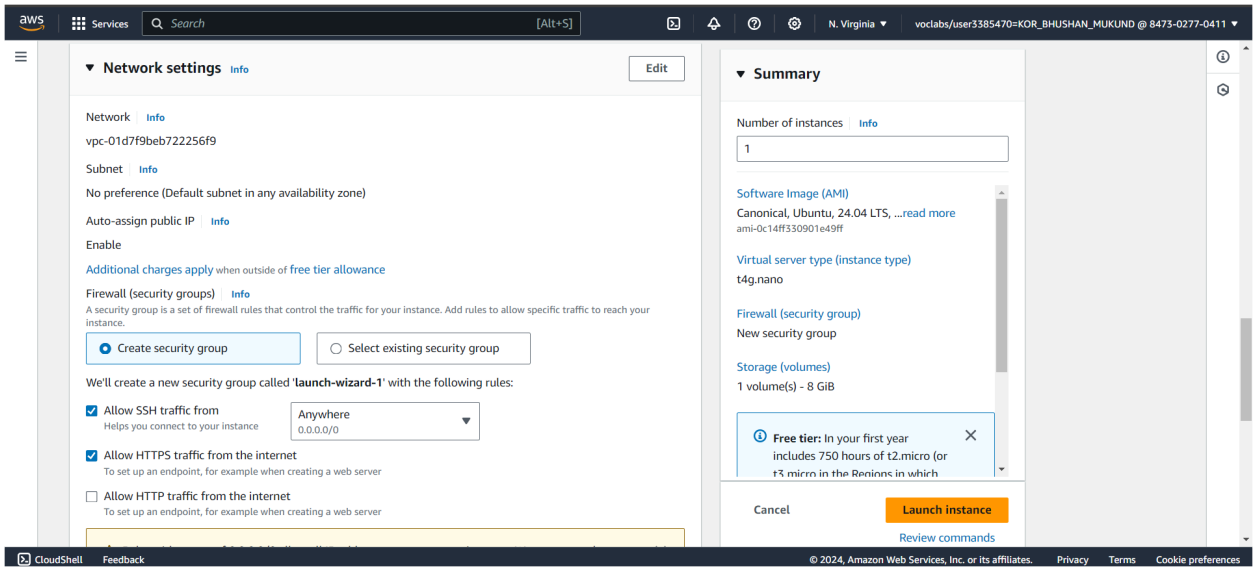
Step 5: Keep instance type by default and key pair as default (vockey).



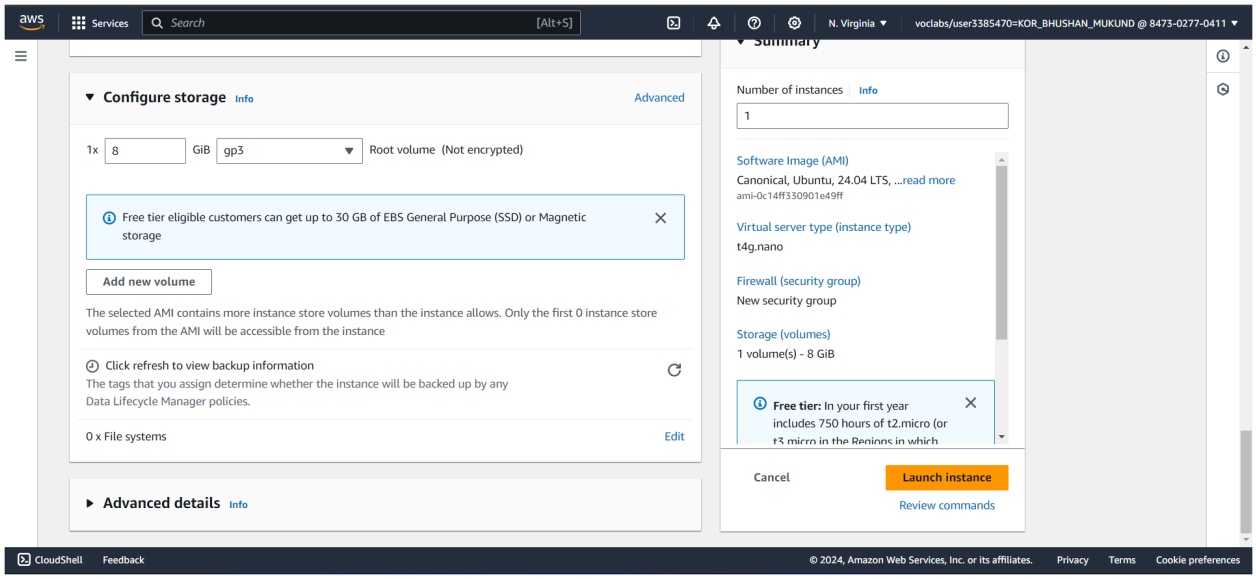
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 6: Select the Create Security Group option and allow 2 permissions refer to screenshot. And if any error occurs allow only 1st permission.



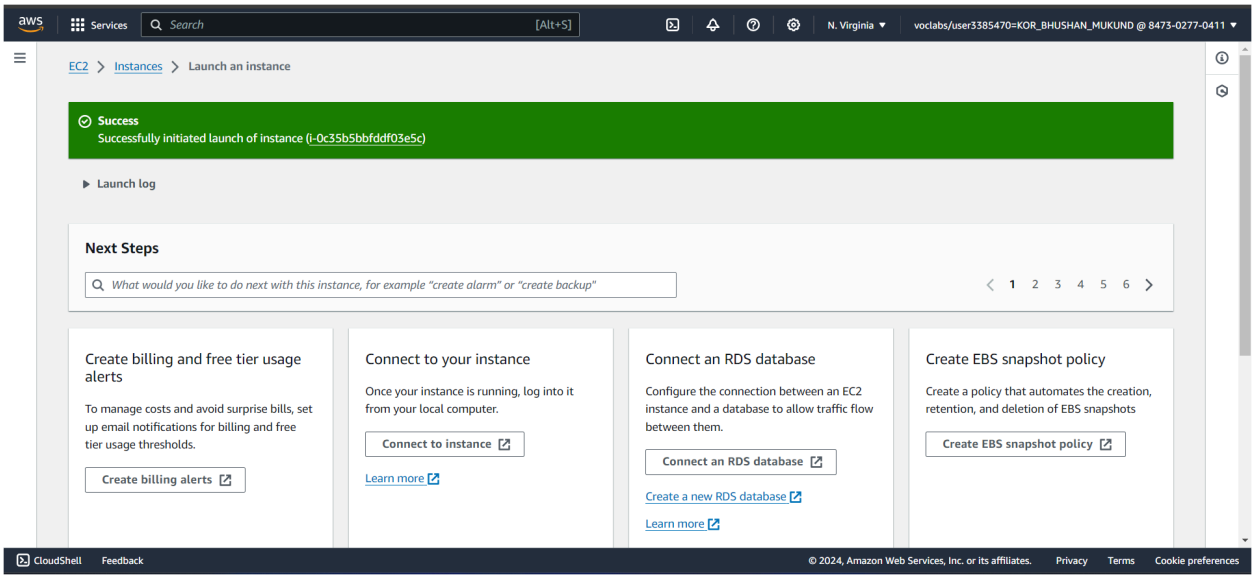
Step 7: Configure storage to 8 GiB and gp3 if any error occurs make it to 10 GiB and gp3.



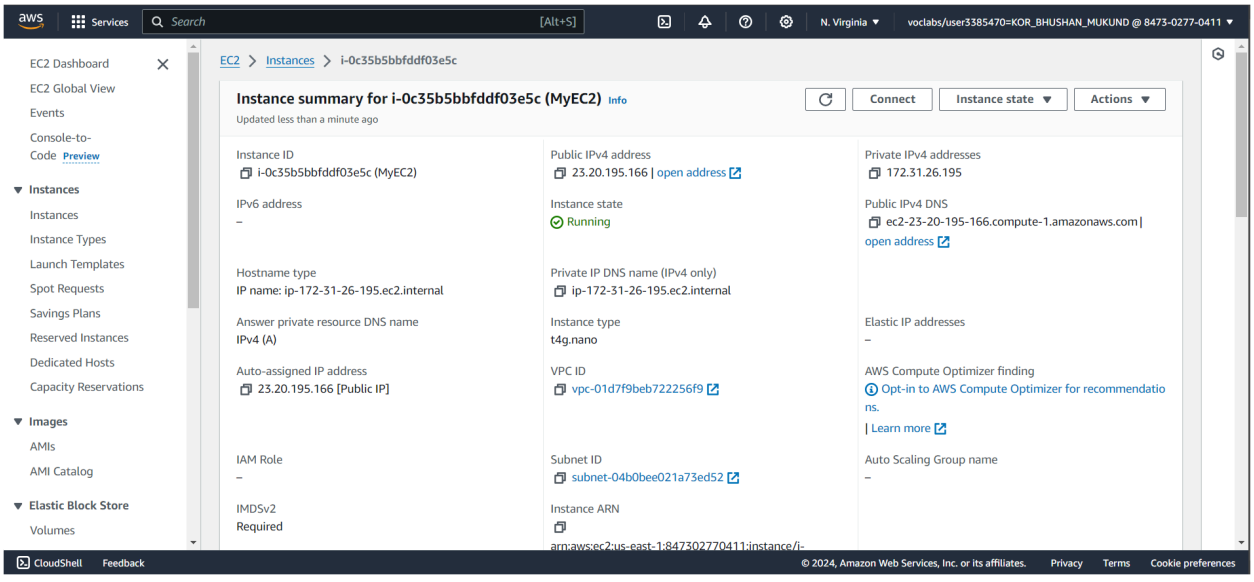
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 10: Done Your EC2 instance is successfully created.

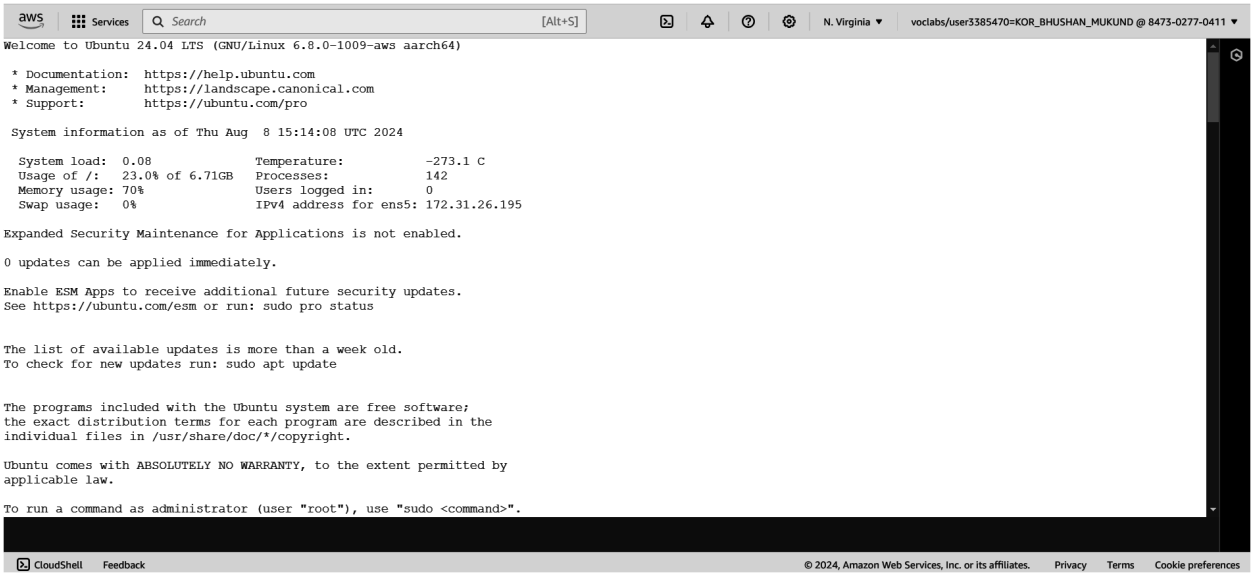


Step 11: Click on the instance ID then click on connect to connect to environment.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



Step 12: Run the following command in Ubuntu (On EC2 Instance).

1. sudo -l
2. sudo apt update
3. uname -a
4. df --help and df
5. ls
6. mkdir test
7. ls
8. cd test
9. touch file1
10. ls
11. touch file2 file3
12. ls
13. rm file1
14. ls
15. rm file*
16. ls
17. cd
18. ls
19. rmdir test
20. ls
21. mkdir test1 test2 test3
22. ls
23. rmdir test*
24. ls

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

25. History
26. top
27. vmstat

```
ubuntu@ip-172-31-26-195:~$ sudo -l
Matching Defaults entries for ubuntu on ip-172-31-26-195:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User ubuntu may run the following commands on ip-172-31-26-195:
    (ALL : ALL) ALL
    (ALL) NOPASSWD: ALL
```

```
ubuntu@ip-172-31-26-195:~$ sudo apt update
Hit:1 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble InRelease
Get:2 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates InRelease [126 kB]
Get:3 http://ports.ubuntu.com/ubuntu-ports noble-security InRelease [126 kB]
Get:4 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports InRelease [126 kB]
Get:5 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/universe arm64 Packages [15.3 MB]
Get:6 http://ports.ubuntu.com/ubuntu-ports noble-security/main arm64 Packages [262 kB]
Get:7 http://ports.ubuntu.com/ubuntu-ports noble-security/main Translation-en [64.4 kB]
Get:8 http://ports.ubuntu.com/ubuntu-ports noble-security/main arm64 Components [5428 B]
Get:9 http://ports.ubuntu.com/ubuntu-ports noble-security/main arm64 c-n-f Metadata [3696 B]
Get:10 http://ports.ubuntu.com/ubuntu-ports noble-security/universe arm64 Packages [244 kB]
Get:11 http://ports.ubuntu.com/ubuntu-ports noble-security/universe Translation-en [108 kB]
Get:12 http://ports.ubuntu.com/ubuntu-ports noble-security/universe arm64 Components [8632 B]
Get:13 http://ports.ubuntu.com/ubuntu-ports noble-security/universe arm64 c-n-f Metadata [9356 B]
Get:14 http://ports.ubuntu.com/ubuntu-ports noble-security/restricted arm64 Packages [206 kB]
Get:15 http://ports.ubuntu.com/ubuntu-ports noble-security/restricted Translation-en [40.7 kB]
Get:16 http://ports.ubuntu.com/ubuntu-ports noble-security/restricted arm64 Components [212 B]
Get:17 http://ports.ubuntu.com/ubuntu-ports noble-security/restricted arm64 c-n-f Metadata [372 B]
Get:18 http://ports.ubuntu.com/ubuntu-ports noble-security/multiverse arm64 Packages [10.1 kB]
Get:19 http://ports.ubuntu.com/ubuntu-ports noble-security/multiverse Translation-en [2808 B]
Get:20 http://ports.ubuntu.com/ubuntu-ports noble-security/multiverse arm64 Components [212 B]
Get:21 http://ports.ubuntu.com/ubuntu-ports noble-security/multiverse arm64 c-n-f Metadata [344 B]
Get:22 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/universe Translation-en [5982 kB]
Get:23 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/universe arm64 Components [3573 kB]
Get:24 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/universe arm64 c-n-f Metadata [295 kB]
Get:25 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/multiverse arm64 Packages [223 kB]
Get:26 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/multiverse Translation-en [118 kB]
Get:27 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/multiverse arm64 Components [31.6 kB]
Get:28 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble/multiverse arm64 c-n-f Metadata [7152 B]
Get:29 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/main arm64 Packages [334 kB]
Get:30 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/main Translation-en [86.2 kB]
Get:31 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/main arm64 c-n-f Metadata [5720 B]
Get:32 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/universe arm64 Packages [315 kB]
Get:33 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/universe Translation-en [135 kB]
Get:34 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/universe arm64 Components [45.0 kB]
Get:35 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/universe arm64 c-n-f Metadata [12.5 kB]
Get:36 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/restricted arm64 Packages [237 kB]
Get:37 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/restricted Translation-en [46.4 kB]
Get:38 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/restricted arm64 c-n-f Metadata [368 B]
Get:39 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/multiverse arm64 Packages [10.1 kB]
Get:40 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/multiverse Translation-en [3608 B]
Get:41 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/multiverse arm64 Components [212 B]
Get:42 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-updates/multiverse arm64 c-n-f Metadata [340 B]
Get:43 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/main arm64 Components [208 B]
Get:44 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/main arm64 c-n-f Metadata [112 B]
Get:45 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/universe arm64 Packages [10.3 kB]
Get:46 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/universe Translation-en [10.5 kB]
Get:47 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/universe arm64 Components [17.7 kB]
Get:48 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/universe arm64 c-n-f Metadata [1020 B]
Get:49 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/restricted arm64 Components [216 B]
Get:50 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/restricted arm64 c-n-f Metadata [116 B]
Get:51 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/multiverse arm64 Components [212 B]
Get:52 http://us-east-1.ec2.ports.ubuntu.com/ubuntu-ports noble-backports/multiverse arm64 c-n-f Metadata [116 B]
Fetched 28.2 MB in 13s (2122 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
47 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

```
ubuntu@ip-172-31-26-195:~$ uname -a
Linux ip-172-31-26-195 6.8.0-1009-aws #9-Ubuntu SMP Fri May 17 20:15:49 UTC 2024 aarch64 aarch64 aarch64 GNU/Linux
ubuntu@ip-172-31-26-195:~$
```

```
ubuntu@ip-172-31-26-195:~$ df --help
Usage: df [OPTION]... [FILE]...
Show information about the file system on which each FILE resides,
or all file systems by default.

Mandatory arguments to long options are mandatory for short options too.
  -a, --all             include pseudo, duplicate, inaccessible file systems
  -B, --block-size=SIZE  scale sizes by SIZE before printing them; e.g.,
                           '-BM' prints sizes in units of 1,048,576 bytes;
                           see SIZE format below
  -h, --human-readable  print sizes in powers of 1024 (e.g., 1023M)
  -H, --si              print sizes in powers of 1000 (e.g., 1.1G)
  -i, --inodes          list inode information instead of block usage
  -k                    like --block-size=1K
  -l, --local           limit listing to local file systems
      --no-sync         do not invoke sync before getting usage info (default)
      --output[=FIELD_LIST]  use the output format defined by FIELD_LIST,
                               or print all fields if FIELD_LIST is omitted.
  -P, --portability     use the POSIX output format
      --sync            invoke sync before getting usage info
      --total           elide all entries insignificant to available space,
                          and produce a grand total
  -t, --type=TYPE       limit listing to file systems of type TYPE
  -T, --print-type      print file system type
  -x, --exclude-type=TYPE   limit listing to file systems not of type TYPE
  -v                    (ignored)
      --help        display this help and exit
      --version     output version information and exit

Display values are in units of the first available SIZE from --block-size,
and the DF_BLOCK_SIZE, BLOCK_SIZE and BLOCKSIZE environment variables.
Otherwise, units default to 1024 bytes (or 512 if POSIXLY_CORRECT is set).
```

```
Display values are in units of the first available SIZE from --block-size,
and the DF_BLOCK_SIZE, BLOCK_SIZE and BLOCKSIZE environment variables.
Otherwise, units default to 1024 bytes (or 512 if POSIXLY_CORRECT is set).

The SIZE argument is an integer and optional unit (example: 10K is 10*1024).
Units are K,M,G,T,P,E,Z,Y,R,Q (powers of 1024) or KB,MB,... (powers of 1000).
Binary prefixes can be used, too: KiB=K, MiB=M, and so on.

FIELD_LIST is a comma-separated list of columns to be included.  Valid
field names are: 'source', 'fstype', 'itotal', 'iused', 'iavail', 'ipcent',
'size', 'used', 'avail', 'pcent', 'file' and 'target' (see info page).

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Report any translation bugs to <https://translationproject.org/team/>
Full documentation <https://www.gnu.org/software/coreutils/df>
or available locally via: info '(coreutils) df invocation'
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
ubuntu@ip-172-31-26-195:~$ mkdir test
ubuntu@ip-172-31-26-195:~$ ls
test
ubuntu@ip-172-31-26-195:~$ cd test
ubuntu@ip-172-31-26-195:~/test$ touch file1
ubuntu@ip-172-31-26-195:~/test$ ls
file1
ubuntu@ip-172-31-26-195:~/test$ touch file2 file3
ubuntu@ip-172-31-26-195:~/test$ ls
file1  file2  file3
ubuntu@ip-172-31-26-195:~/test$ rm file1
ubuntu@ip-172-31-26-195:~/test$ ls
file2  file3
ubuntu@ip-172-31-26-195:~/test$ rm file*
ubuntu@ip-172-31-26-195:~/test$ ls
ubuntu@ip-172-31-26-195:~/test$ cd
```

```
ubuntu@ip-172-31-26-195:~$ ls
test
ubuntu@ip-172-31-26-195:~$ rmdir test
ubuntu@ip-172-31-26-195:~$ ls
ubuntu@ip-172-31-26-195:~$ mkdir test1 test2 test3
ubuntu@ip-172-31-26-195:~$ ls
test1  test2  test3
ubuntu@ip-172-31-26-195:~$ rmdir test*
ubuntu@ip-172-31-26-195:~$ ls
ubuntu@ip-172-31-26-195:~$ history
```

```
ubuntu@ip-172-31-26-195:~$ df
Filesystem      1K-blocks    Used Available Use% Mounted on
/dev/root         7034376 1831888   5186104  27% /
tmpfs              212128       0    212128   0% /dev/shm
tmpfs               84852    1040     83812   2% /run
tmpfs                5120       0      5120   0% /run/lock
efivarfs              128       4       125   3% /sys/firmware/efi/efivars
/dev/nvme0n1p16    911580   57648    790292   7% /boot
/dev/nvme0n1p15     99791    6475     93317   7% /boot/efi
tmpfs               42424      12     42412   1% /run/user/1000
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

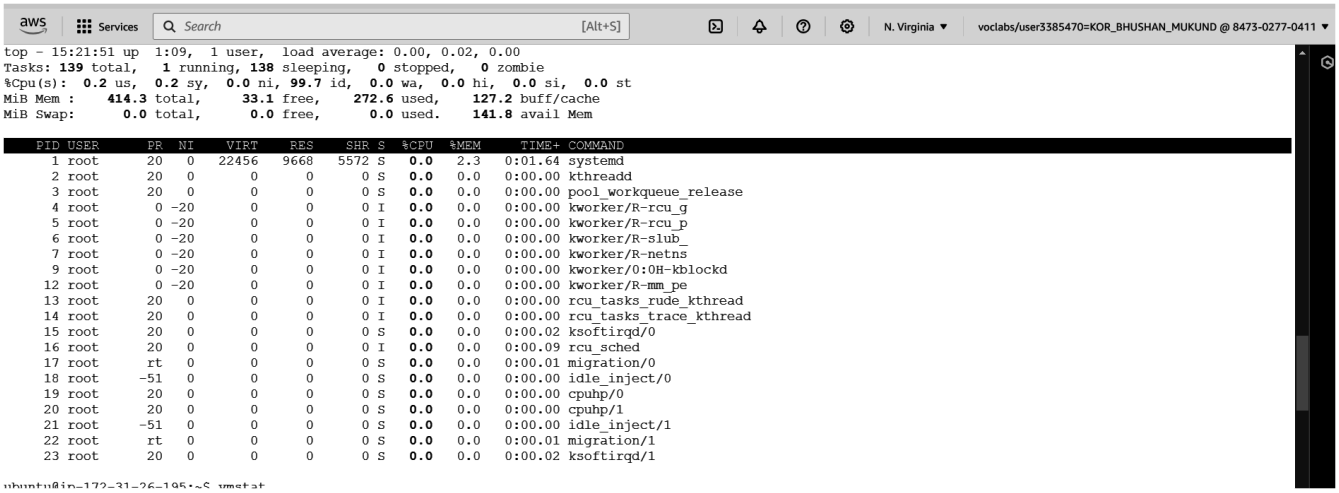
Division: D15C **Roll No: 28**

```
ubuntu@ip-172-31-26-195:~$ history
    1  sudo -l
    2  apt update
    3  sudo apt update
    4  mkdir test
    5  ls
    6  cd test
    7  touch file1
    8  ls
    9  touch file2 file3
   10  ls
   11  rm file1
   12  ls
   13  rm file*
   14  ls
   15  cd
   16  ls
   17  rmdir test
   18  mkdir test1 test2 test3
   19  ls
   20  rmdir test*
   21  ls
   22  history
   23  top
   24  vmstat
   25  df
   26  whatis df
   27  df --help
   28  uname -a
   29  ls
   30  mkdir test
   31  ls
   32  rmdir test
   33  ls
   34  mkdir test1 test2 test3
   35  ls
   36  rmdir test*
   37  ls
   38  history
ubuntu@ip-172-31-26-195:~$
```

```
ubuntu@ip-172-31-26-195:~$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- -------cpu-------
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st gu
 2  0      0  36688   3180 128208    0    0   193   137   43    0  0  0 99  0  0  0
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

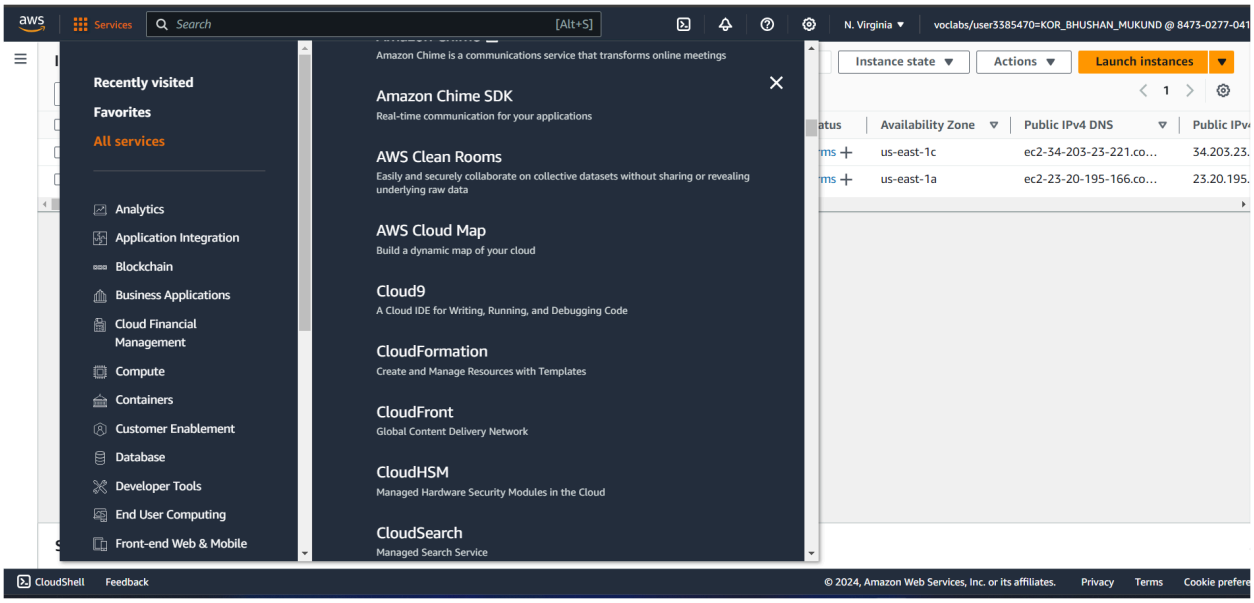
B)Cloud 9 and Cloud 9 IDE

Cloud 9 and Cloud 9 IDE

Cloud 9 is a popular term in English meaning extreme happiness or euphoria. It has no technical association with development tools.

Cloud 9 IDE, on the other hand, is an online integrated development environment (IDE) that allows developers to write, run, and debug code in various programming languages directly in the cloud. It was originally an independent product but is now part of Amazon Web Services (AWS).

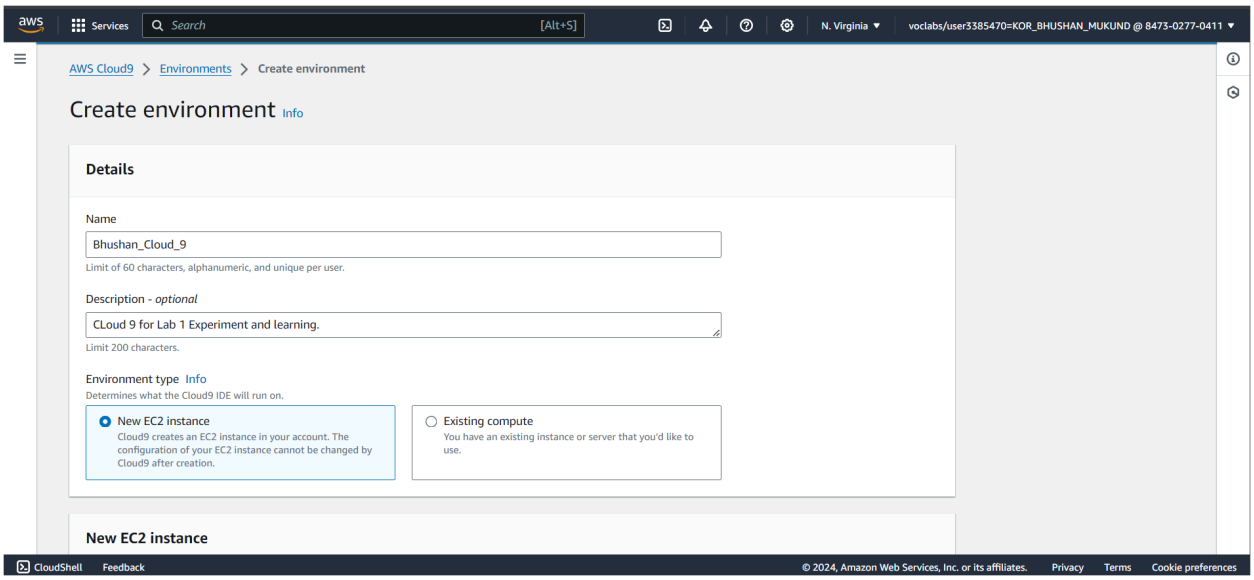
Step 1: Select Cloud 9 from services in AWS.



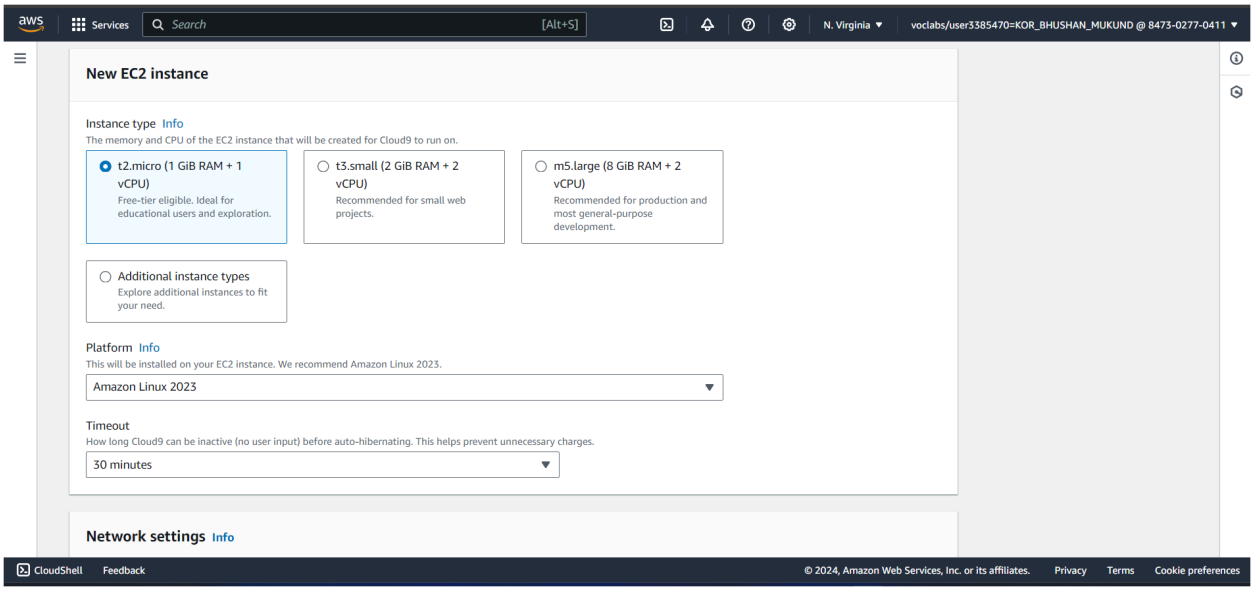
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 2: Now Click on Create Environment and give name to your environment and select new EC2 instance option.



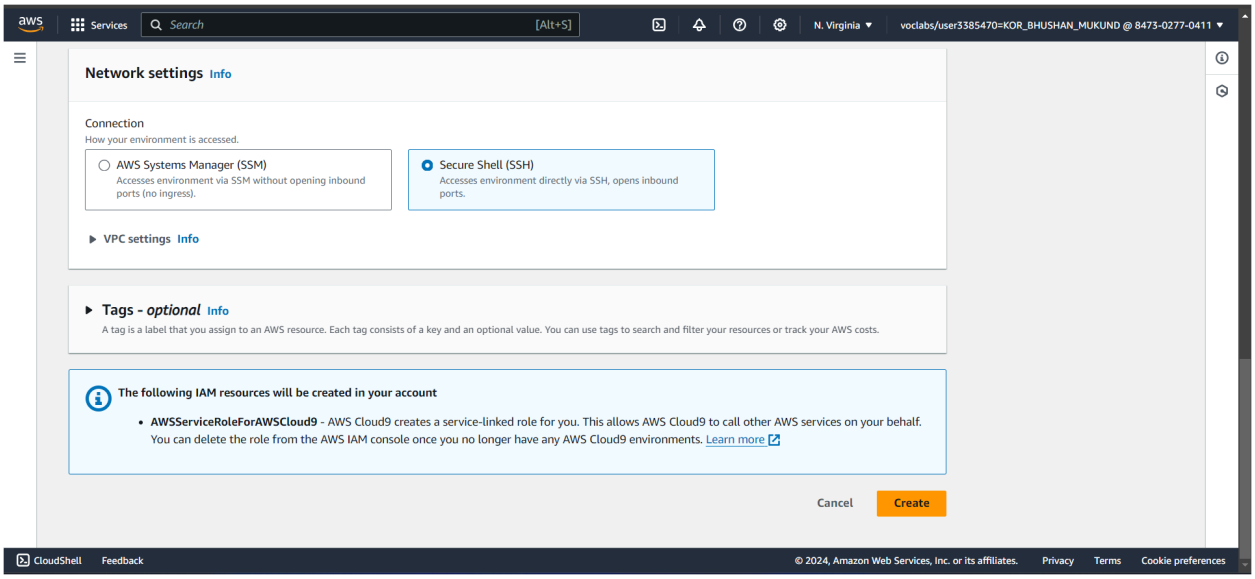
Step 3: Keep other things by default then select platform as Amazon Linux 2023 or latest and time 30 minutes.



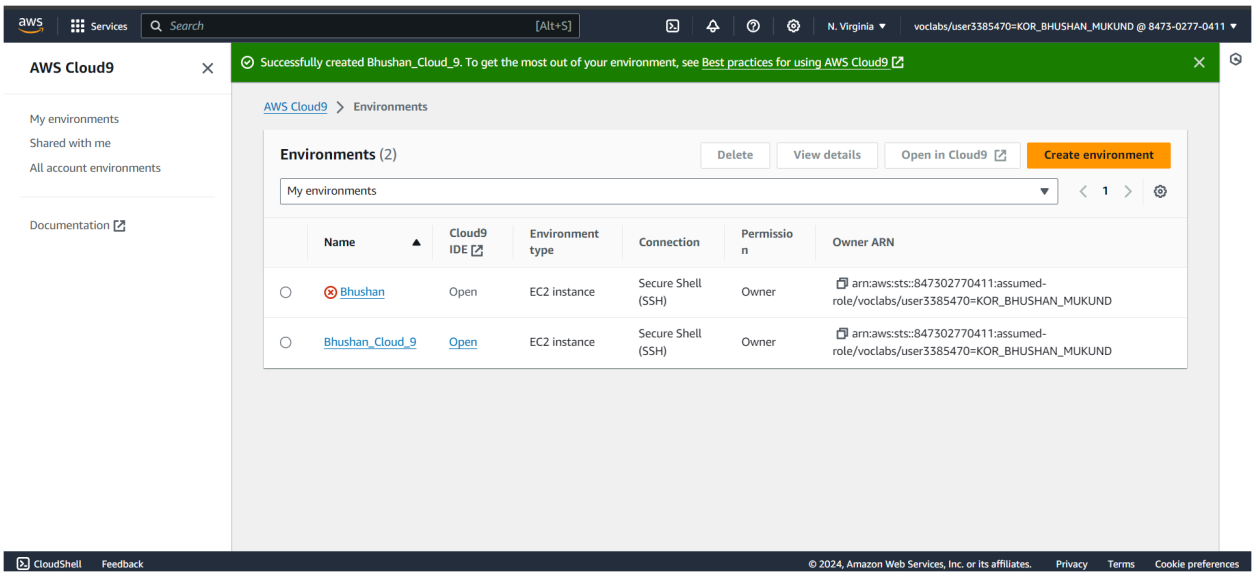
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 4: Select Secure Shell (SSH) and click on Create.

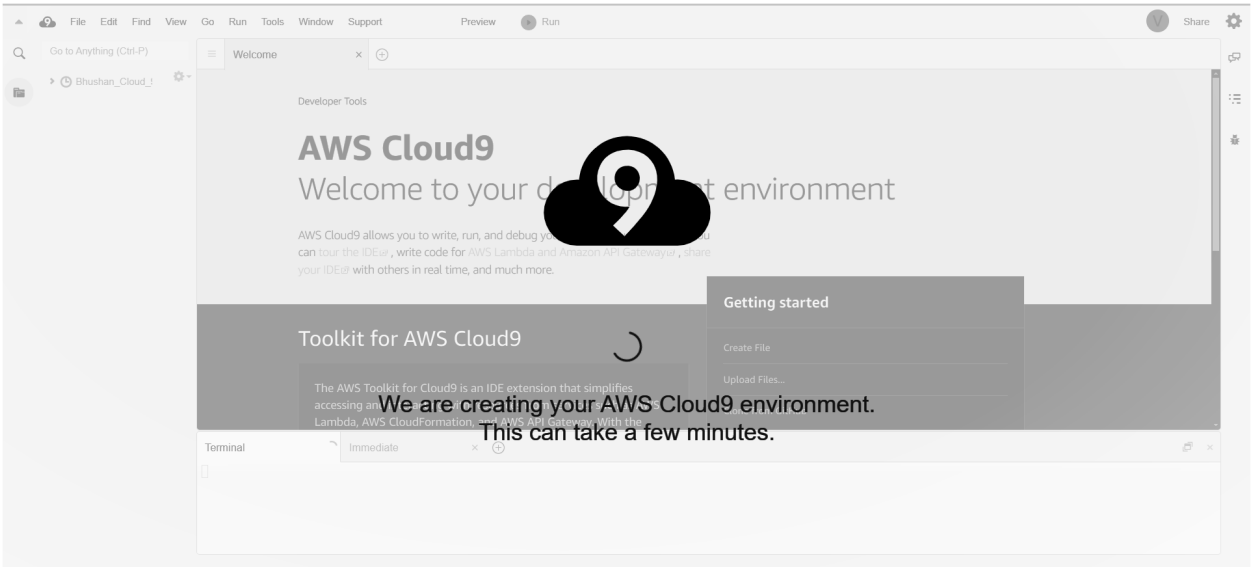


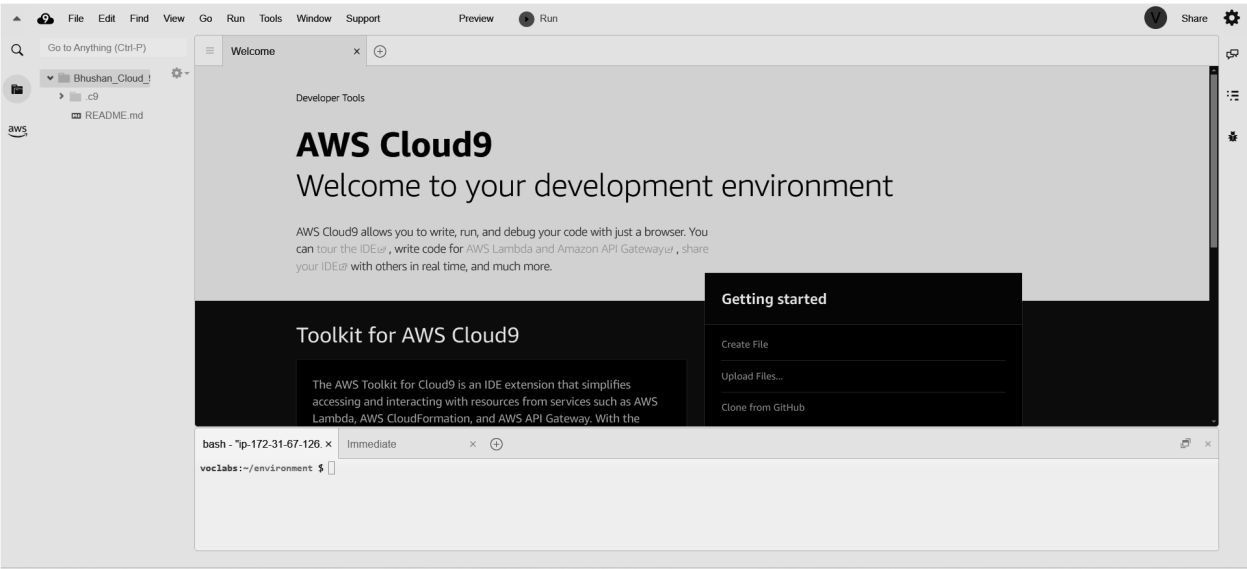
Step 5: Now It will create the environment after that open it.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

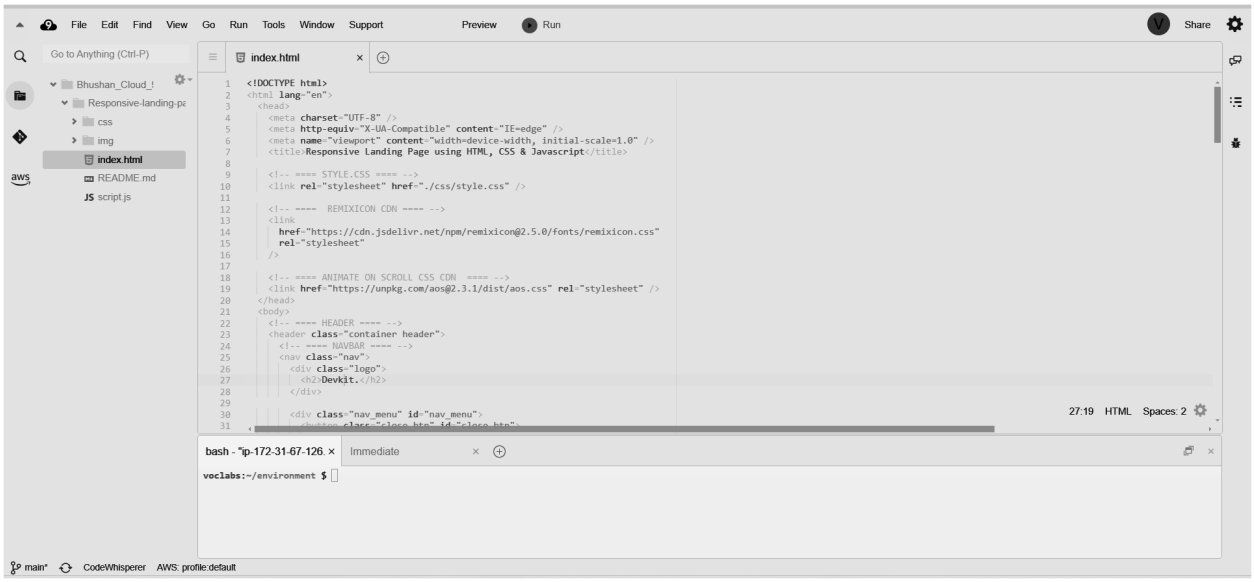


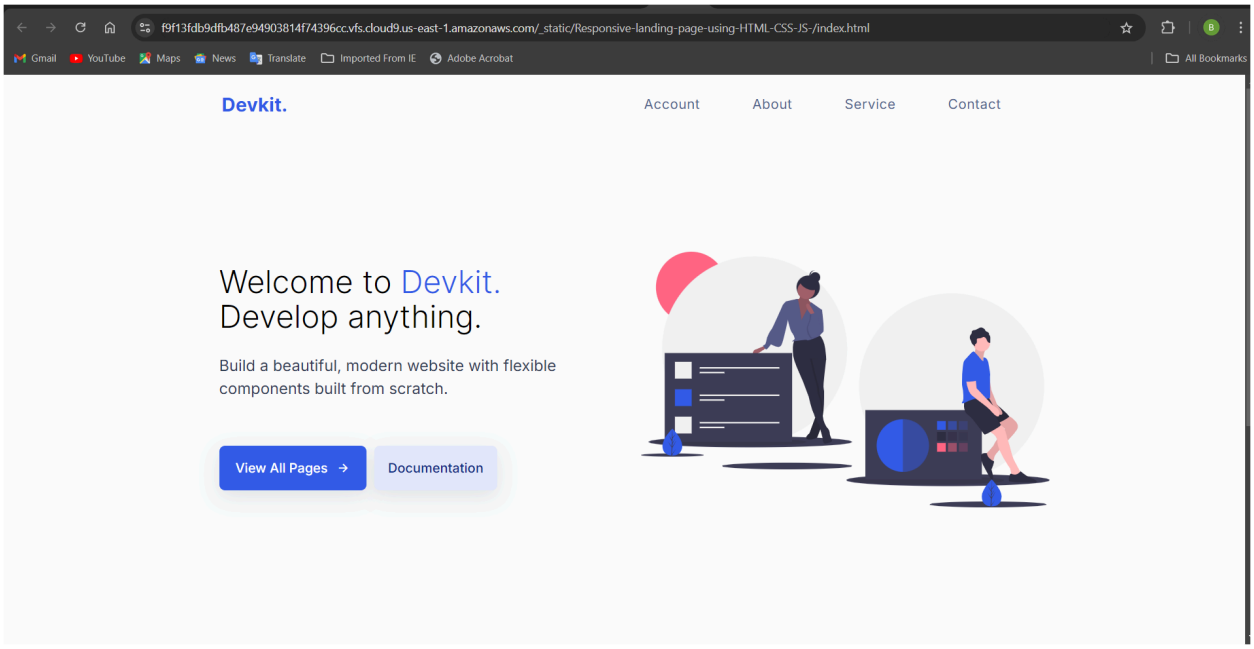


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

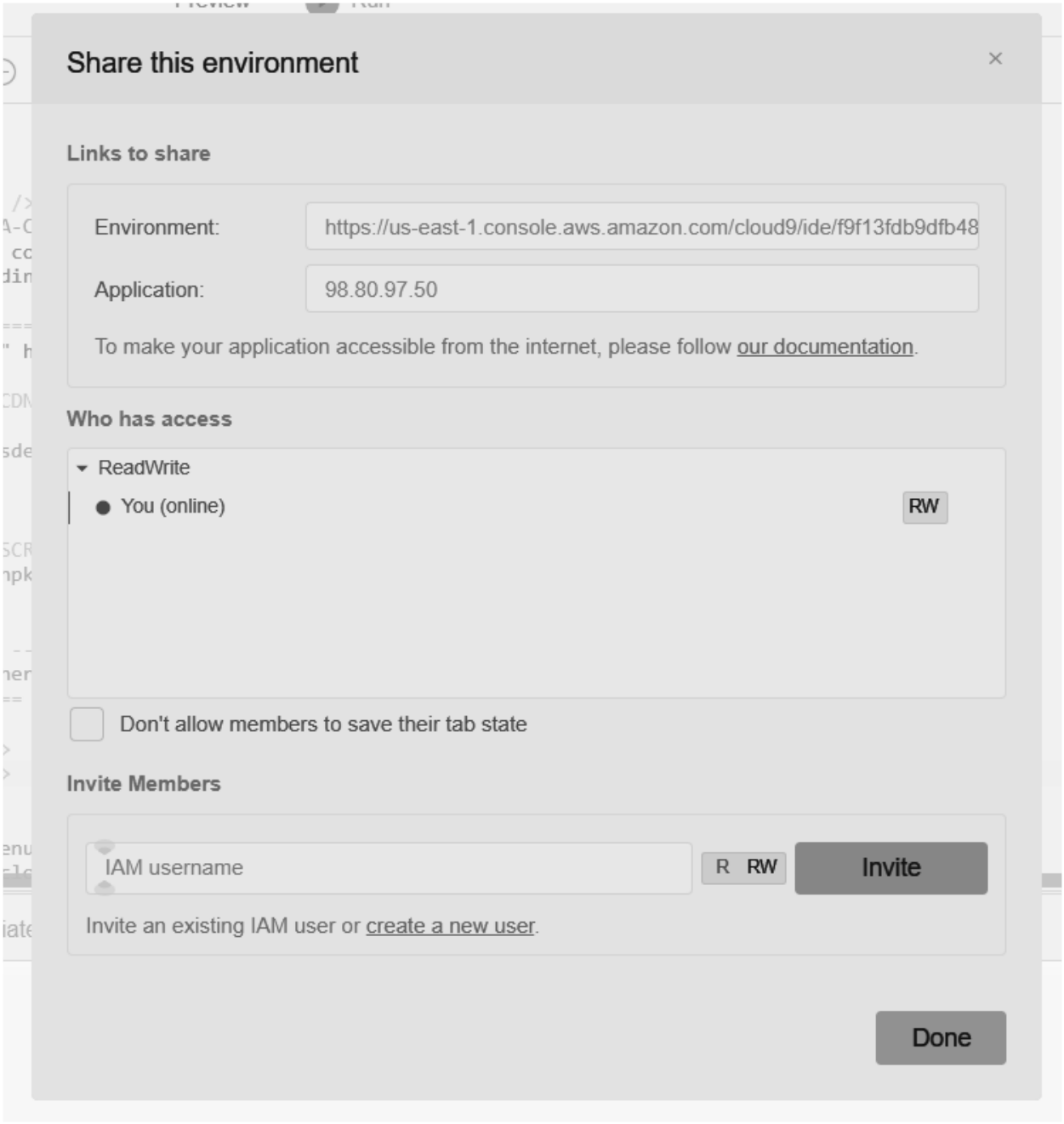
Division: D15C **Roll No: 28**

Step 6: Add some files or create some files and see the preview of it. Here we have created an index.html file.

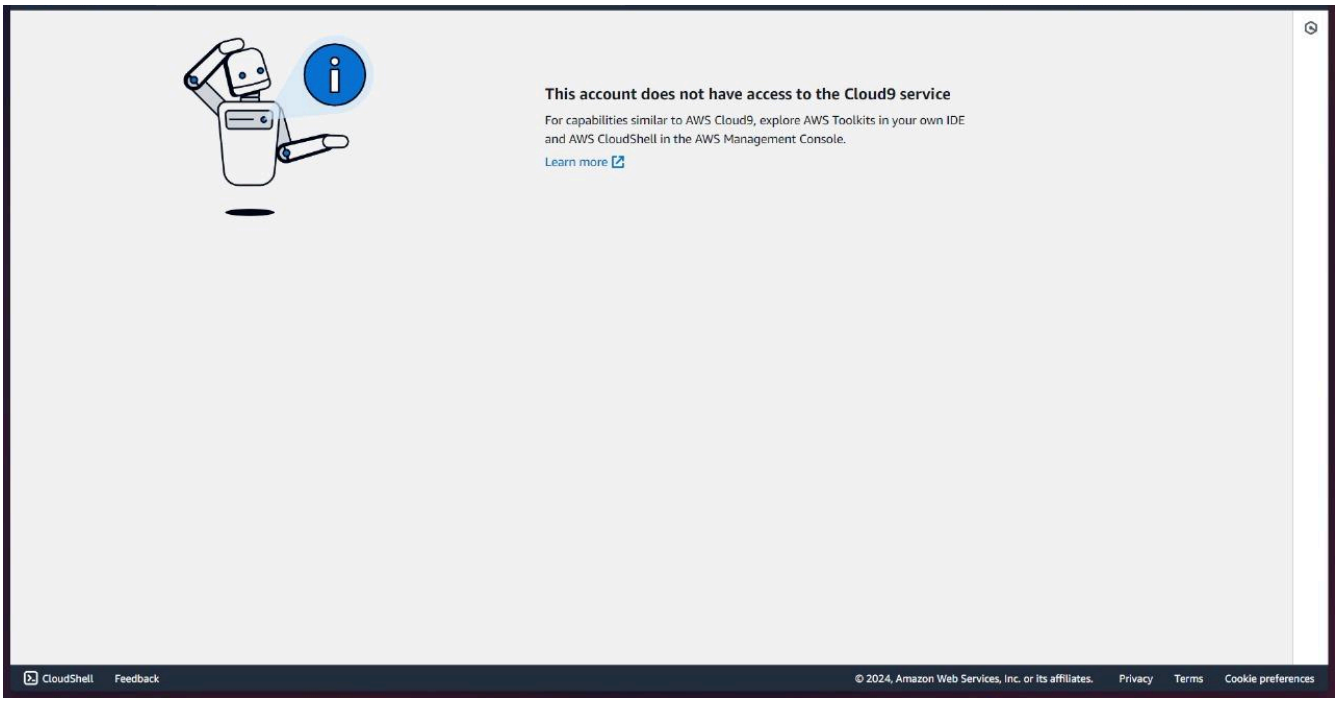


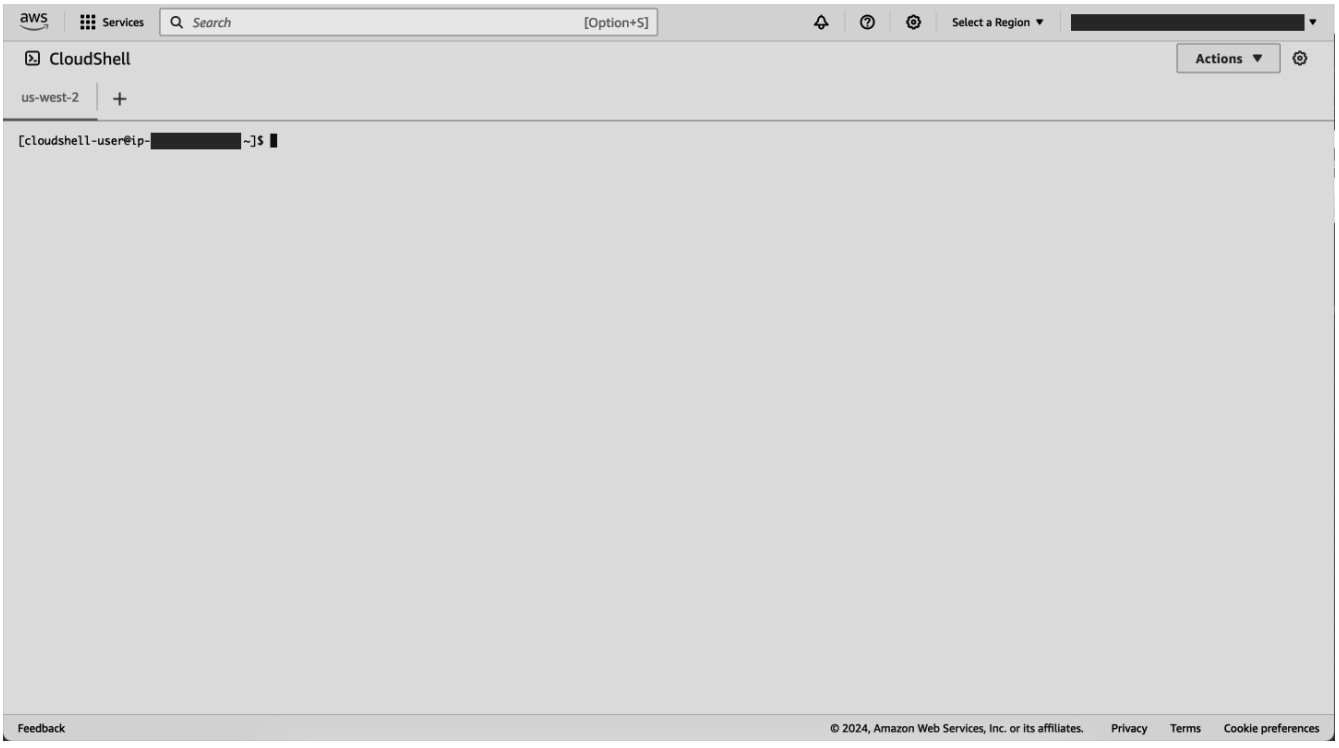


Step 7: Now lets share the file with IAM users.



Step 8: Note That sharing is not possible as we are using AWS academy account in which we do not have access to the IAM policies. And Also 2nd Important thing to note that is AWS has closed Cloud 9 for those who have created their AWS account after 25 Jul 2024. Instead of Cloud 9 AWS has alternative as AWS IDE Toolkits and AWS cloudshell.





Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To Build Your Application using AWS CodeBuild and Deploy on S3 / SEBS using AWS CodePipeline, deploy Sample Application on EC2 instance using AWS CodeDeploy.

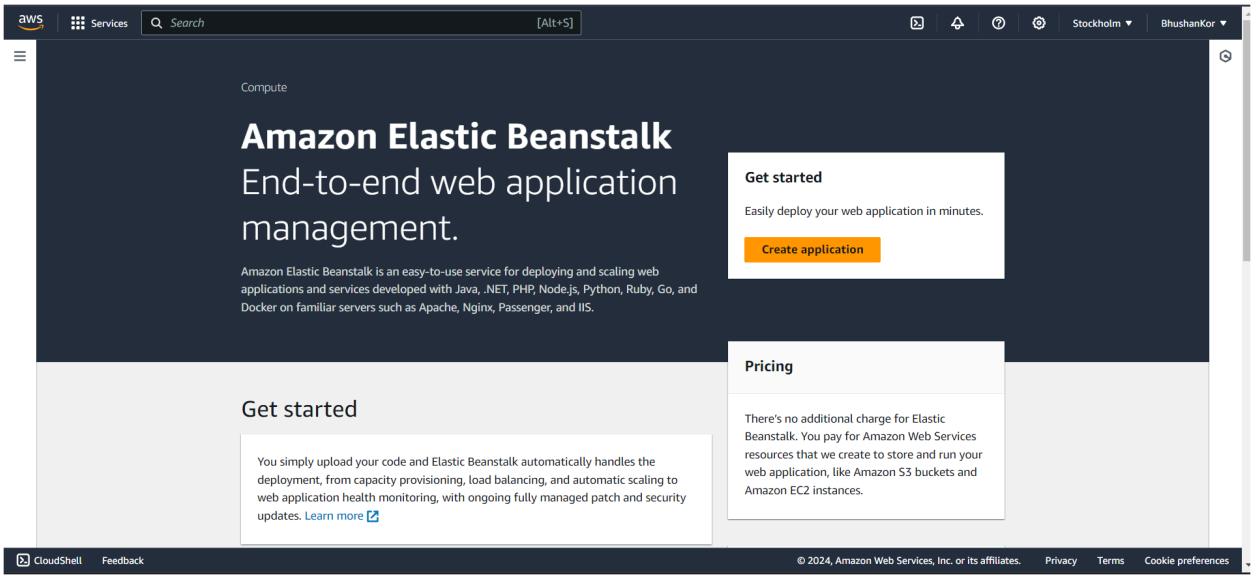
Elastic Beanstalk:
Amazon Elastic Beanstalk is a fully managed AWS service that streamlines the deployment and scaling of web applications. Developers simply upload their code, and Elastic Beanstalk automatically handles tasks like resource provisioning, load balancing, auto-scaling, and application health monitoring. It supports multiple programming languages and frameworks, offering flexibility and control over the underlying AWS resources if needed. This enables rapid application deployment without the complexities of managing infrastructure.

CodePipeline:
A CodePipeline is an AWS service that automates the build, test, and deploy phases of your release process. It enables continuous integration and continuous delivery (CI/CD) by defining the workflow of code changes from source to production. CodePipeline integrates with various AWS services and third-party tools to streamline and accelerate the release of new features.

Prerequisites: Before you start go to the end of this document and do the steps of New learning because it is necessary for step 5 and codePipeline.

A)Elastic Beanstalk

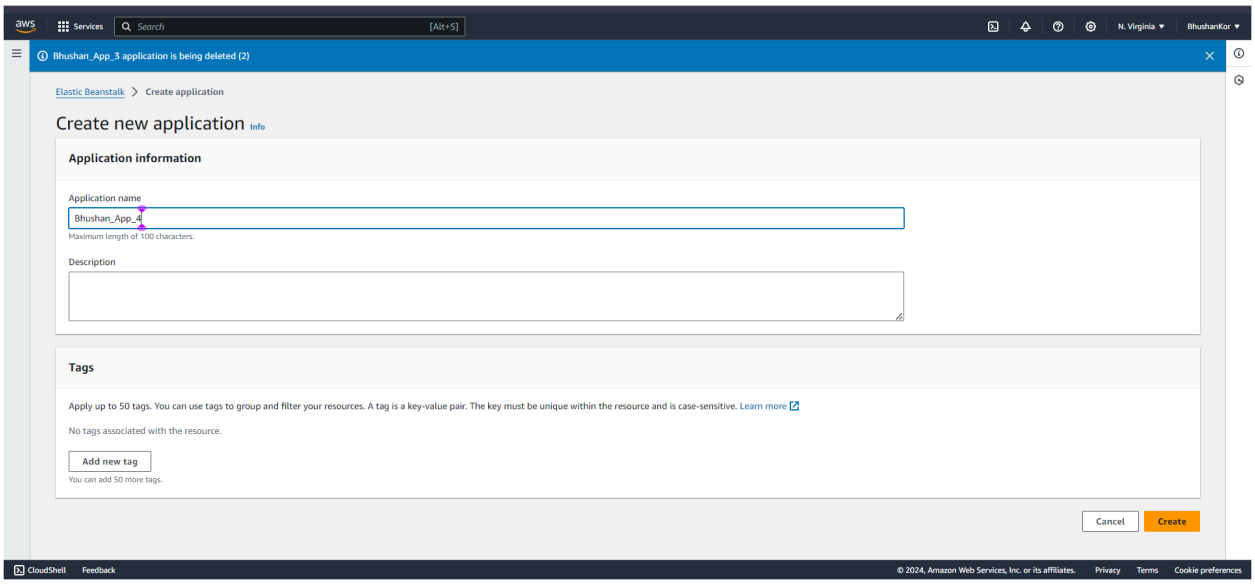
Step 1: Open Your **personal AWS** Account Because Elastic Beanstalk can be created in AWS Academy but connecting GitHub with the code pipeline requires an IAM policy that is not present in the AWS Academy account.
In services search for Elastic Beanstalk and click on it.



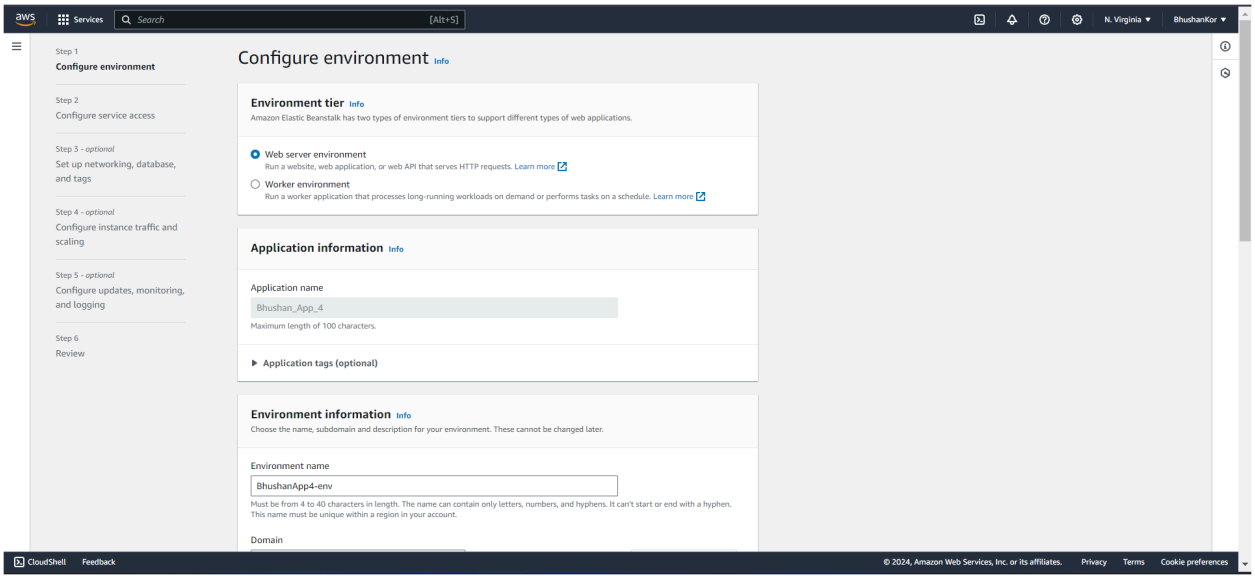
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 2: Click on Create a new application and give a name to your application.



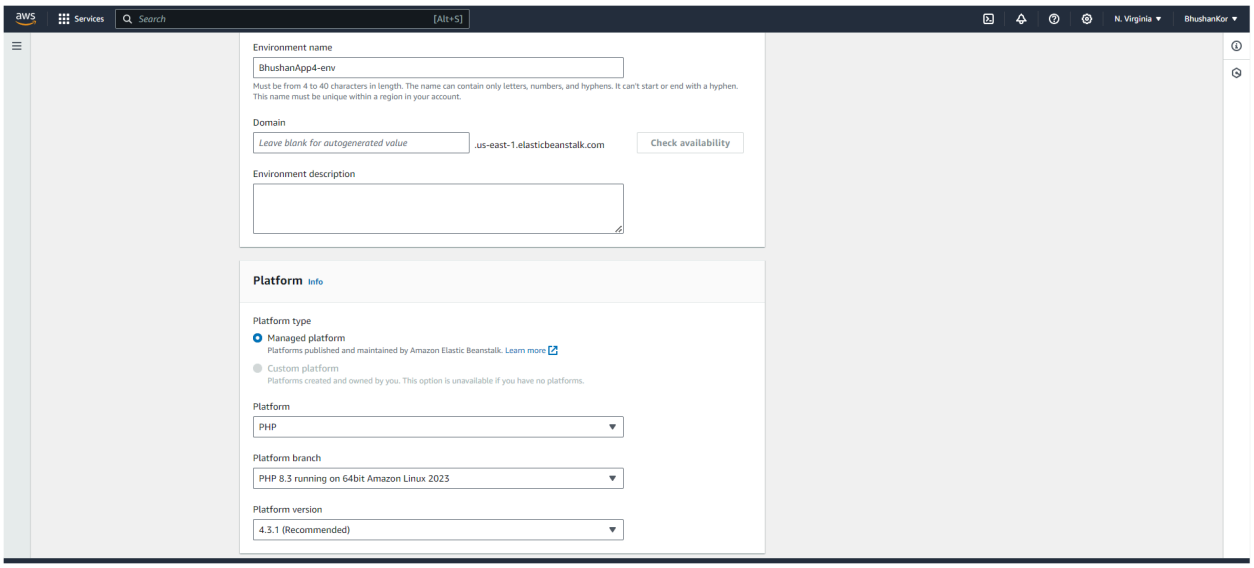
Step 3: After creation of the application click on Create Environment and select Web Server Environment.

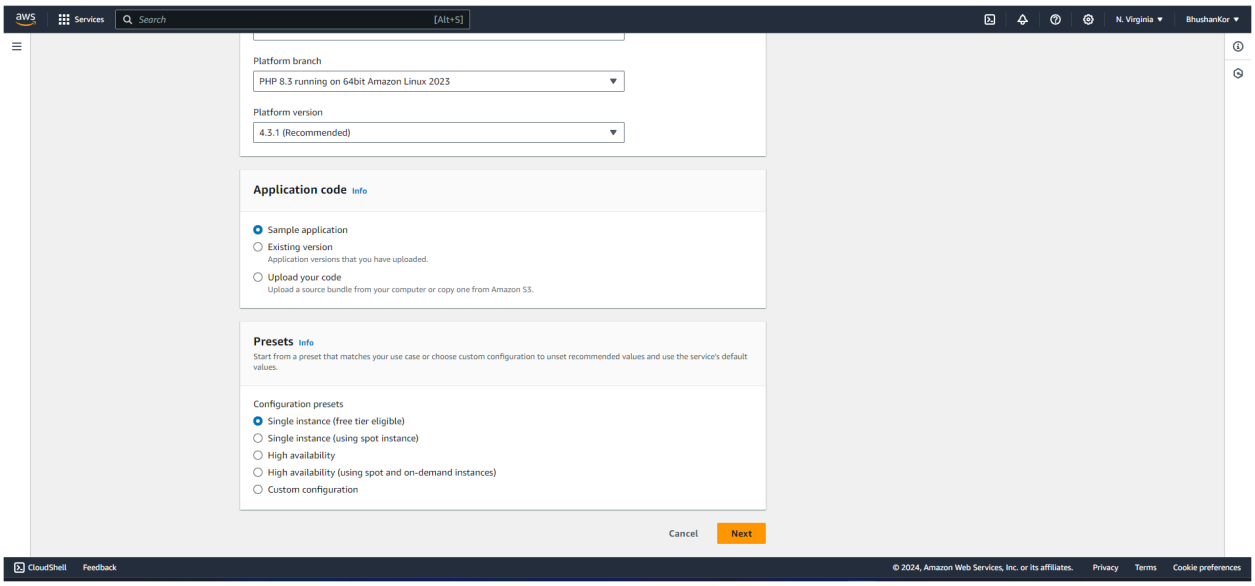


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 4: Select the platform as PHP and application code to sample or if you have you can upload and keep others to default.



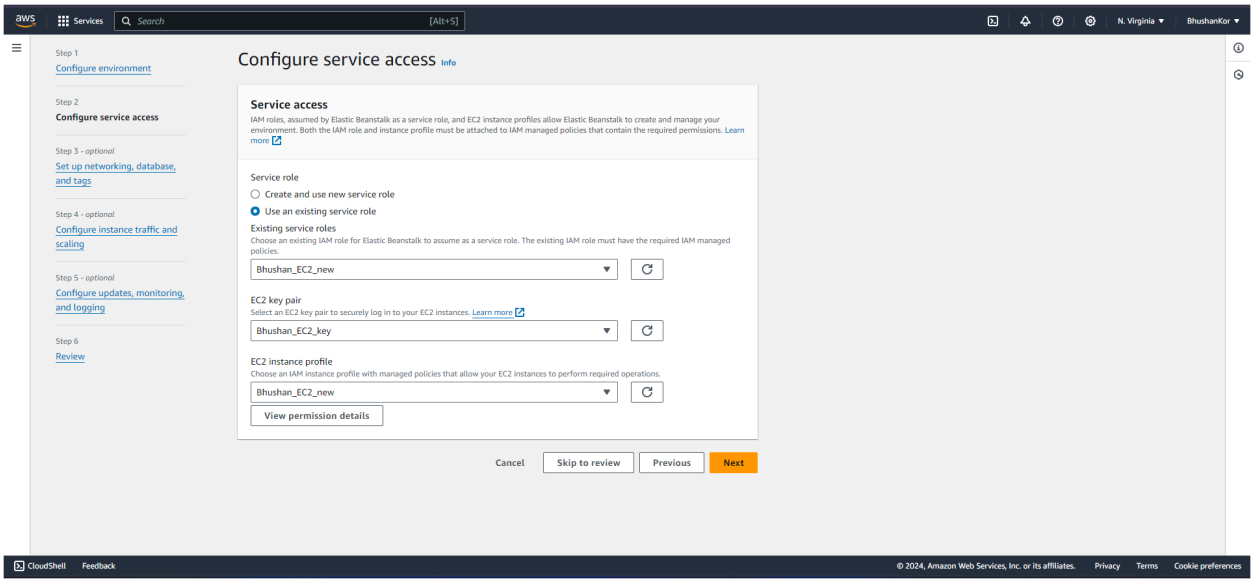


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

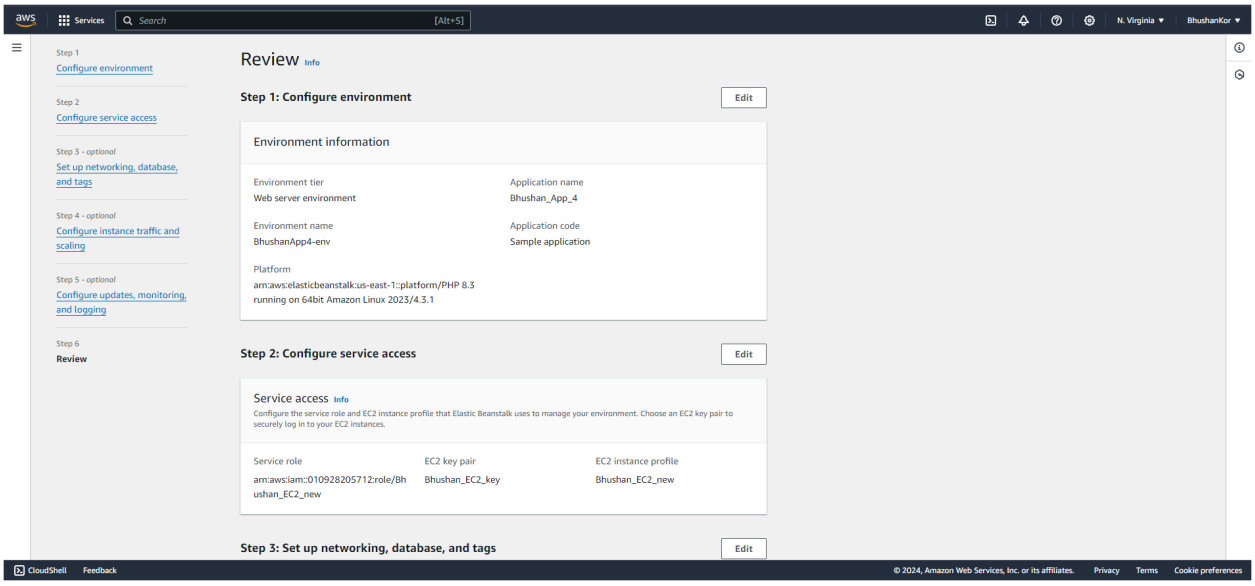
Division: D15C **Roll No: 28**

Step 5: Select the use an existing service role option because create and use new service role might not work according to the new policy of AWS (Refer to the New learnings of this document).

Select the existing roles and instances and keys.

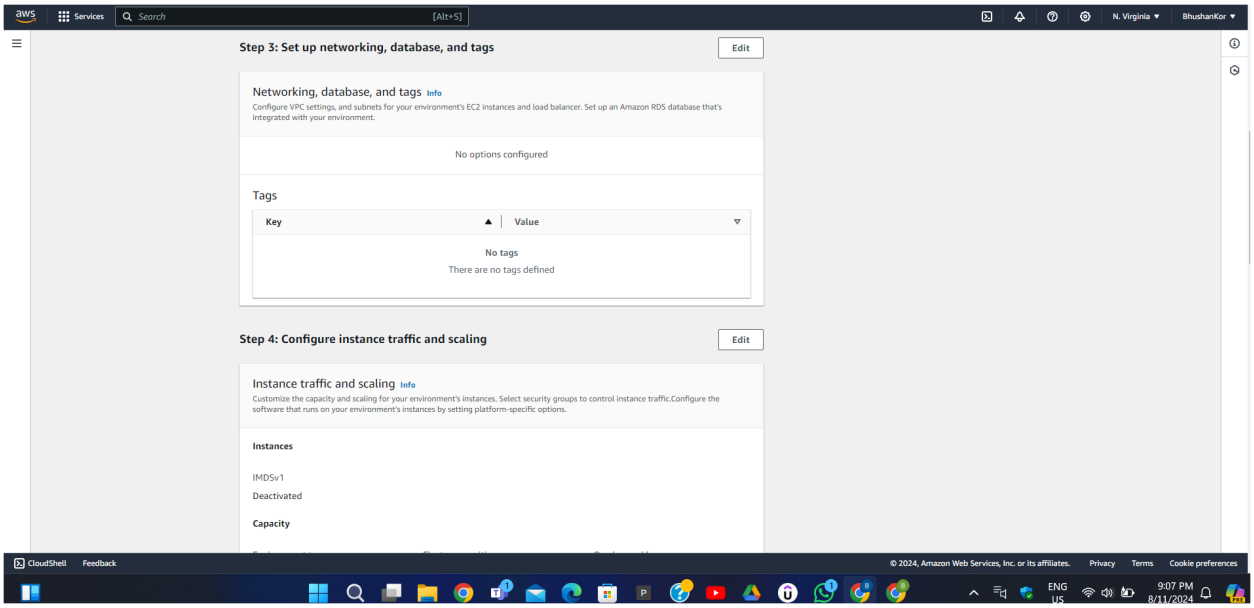


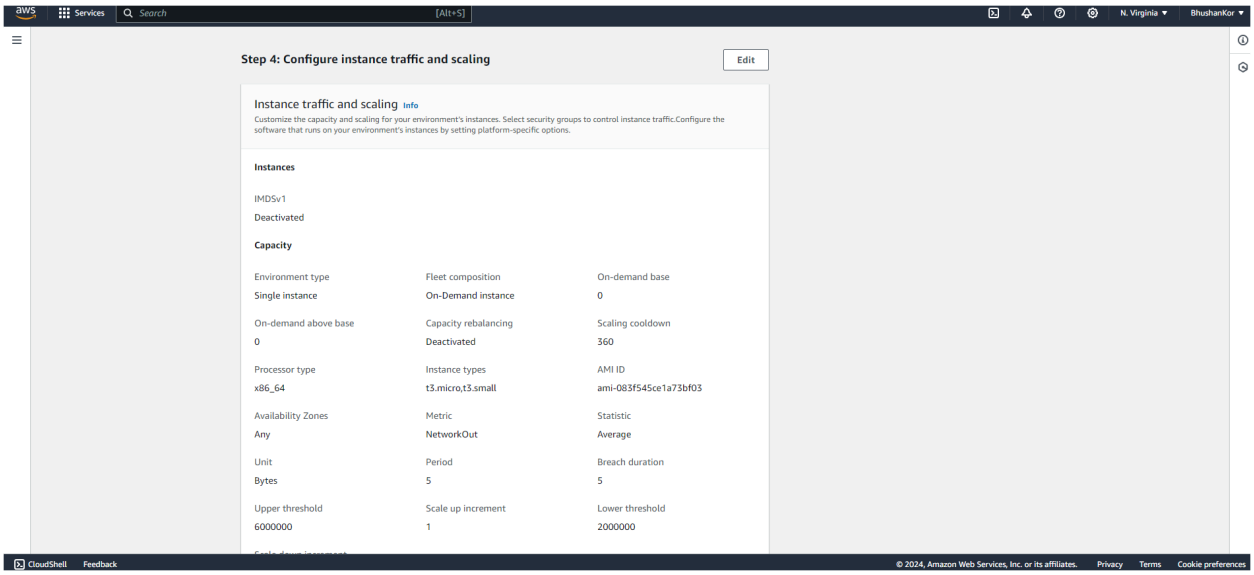
Step 6: Review and click on submit.

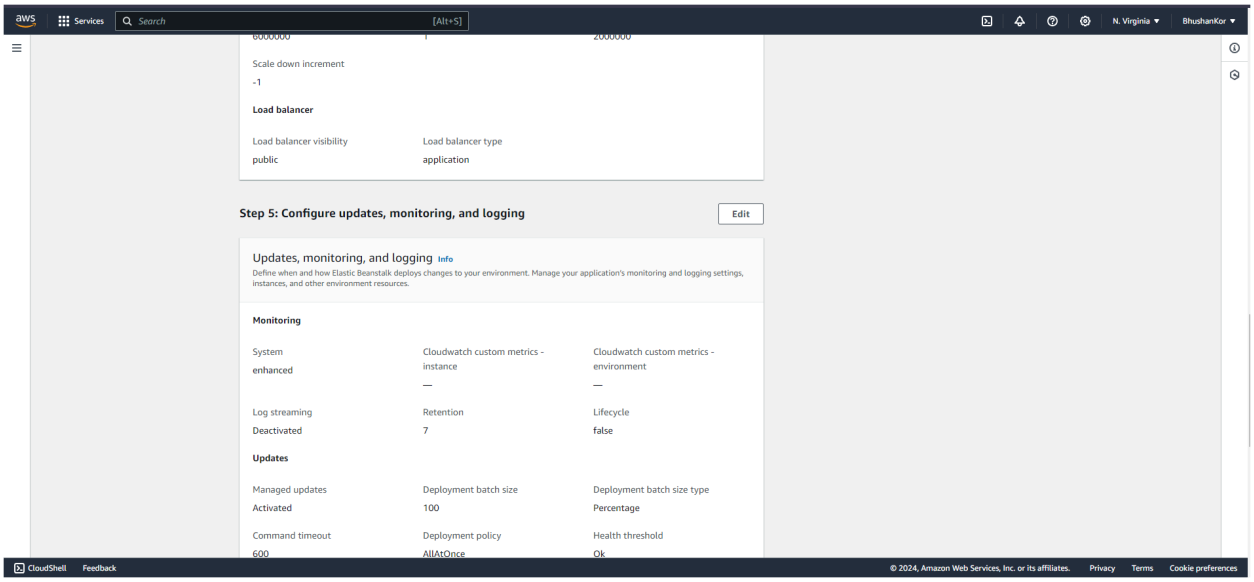


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

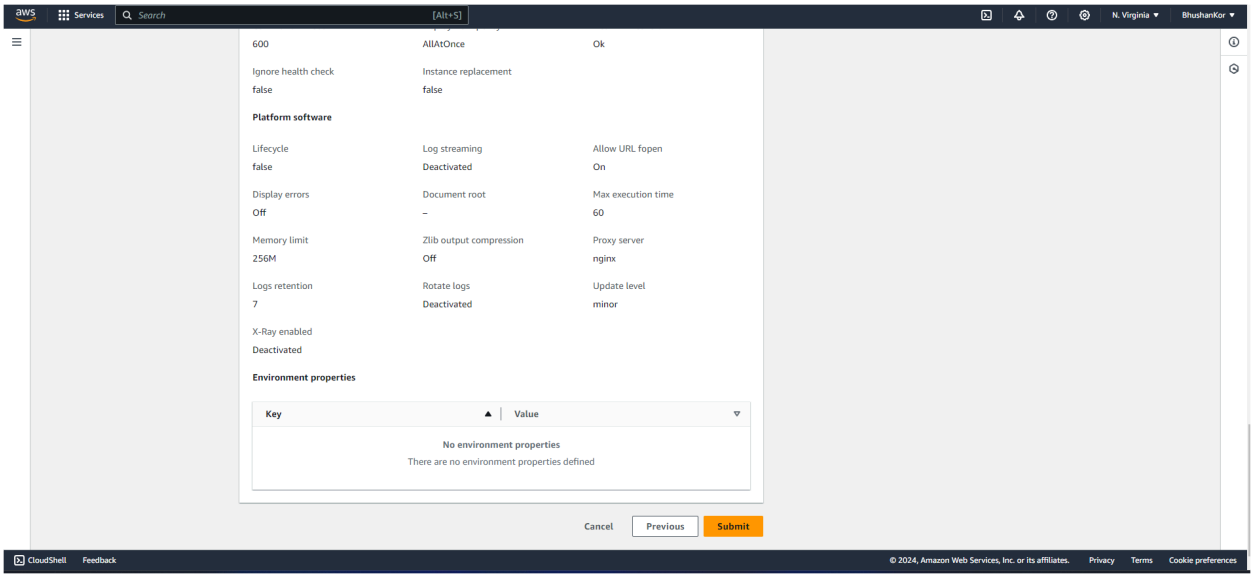




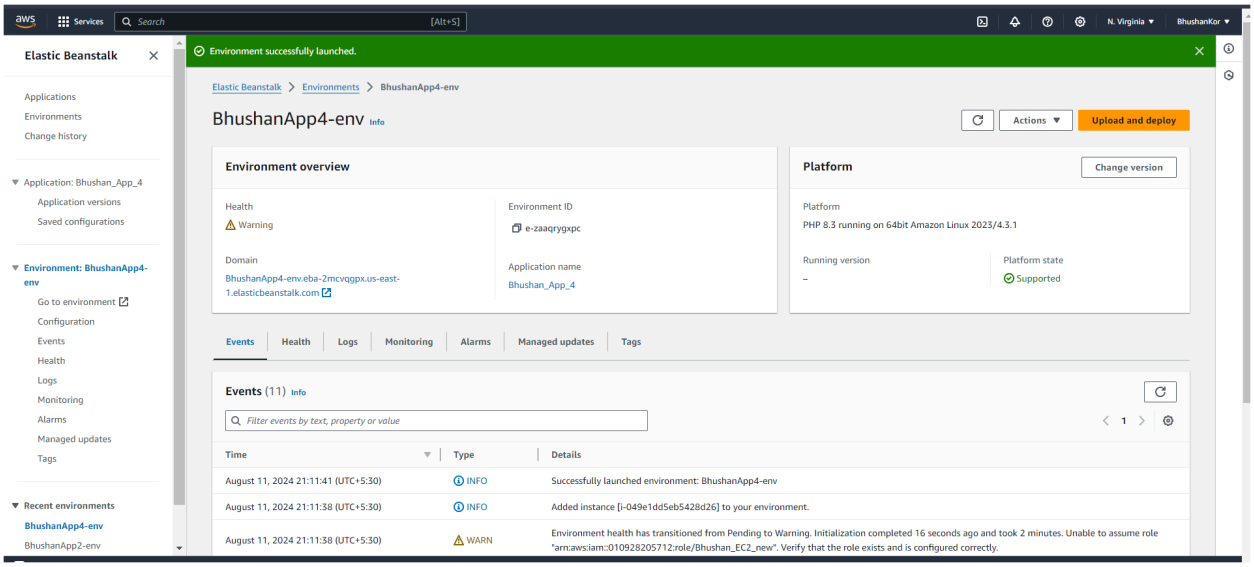


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

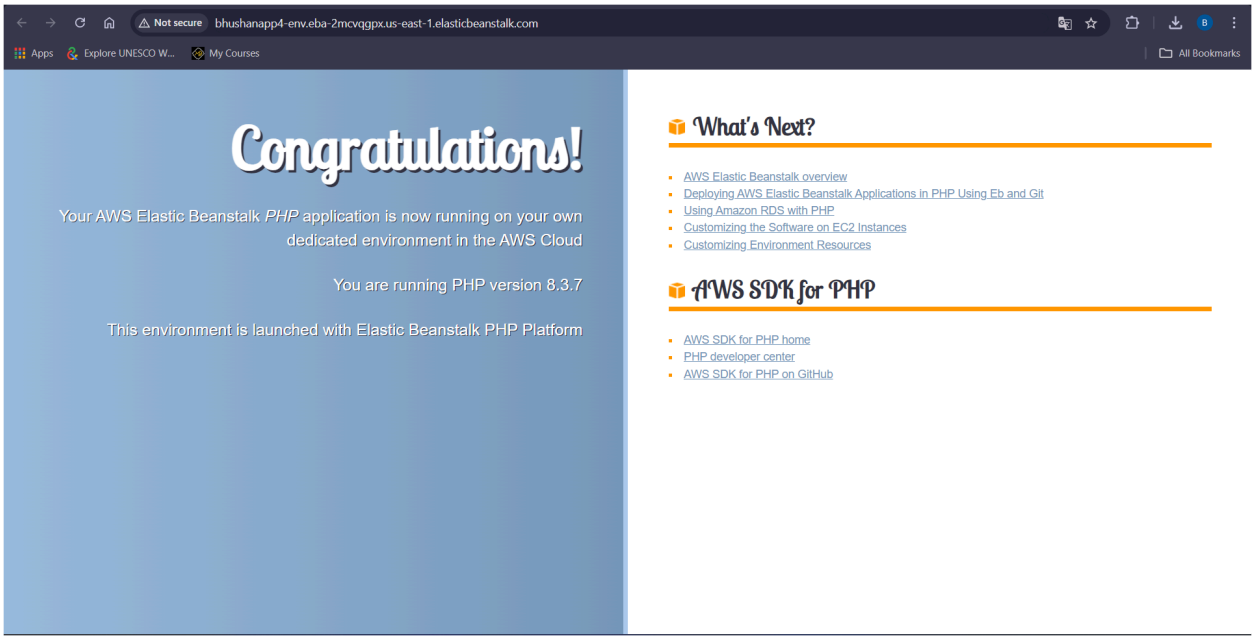


Step 7: Done Elastic BeanStalk environment is created successfully, click on the link to view the preview of your sample code or uploaded code.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

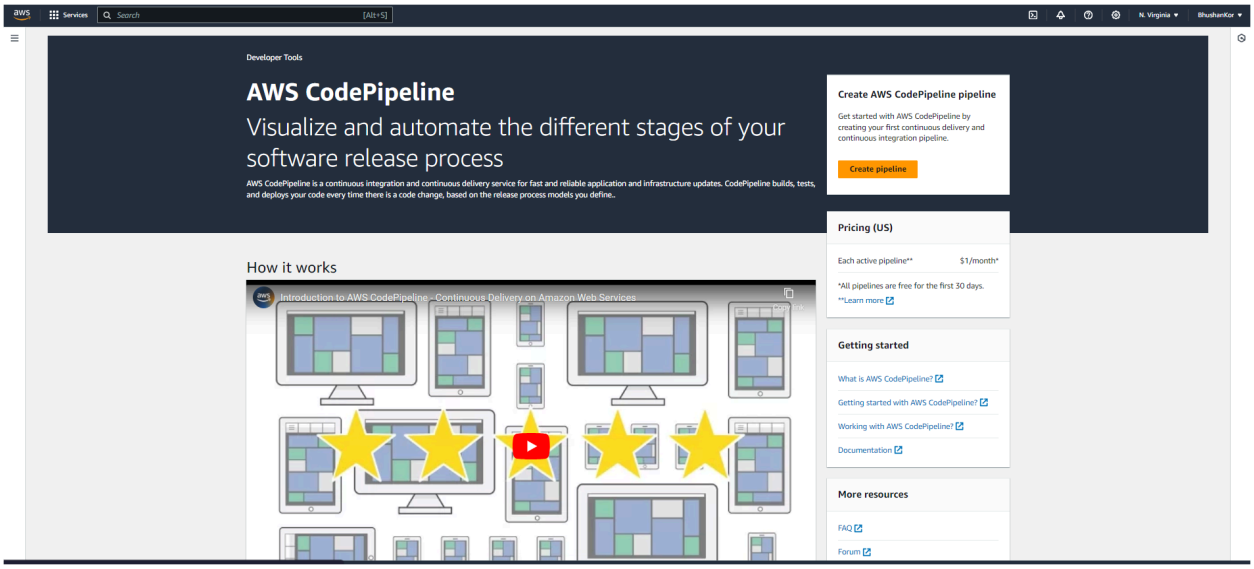


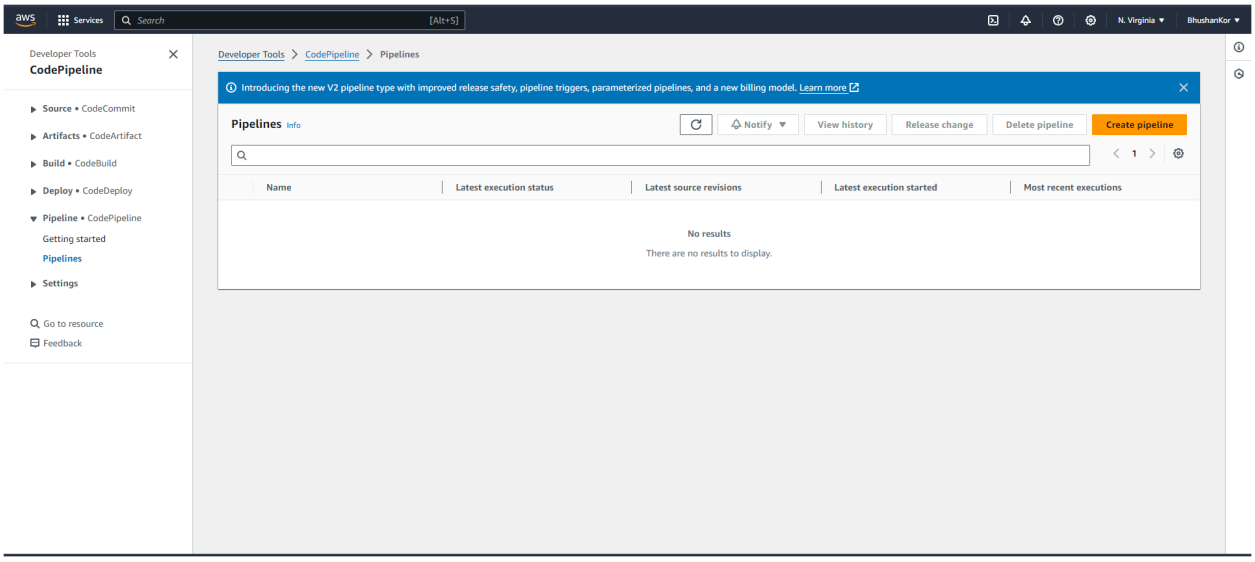
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

B)CodePipline

Step 1: Search for codePipeline in services and click on create pipeline.

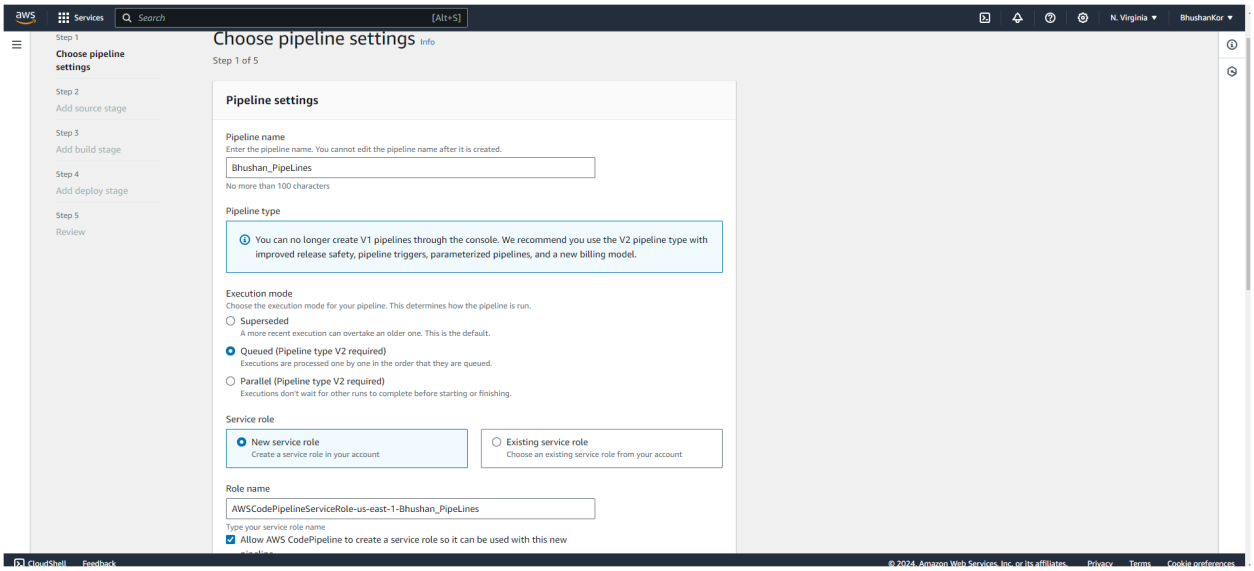


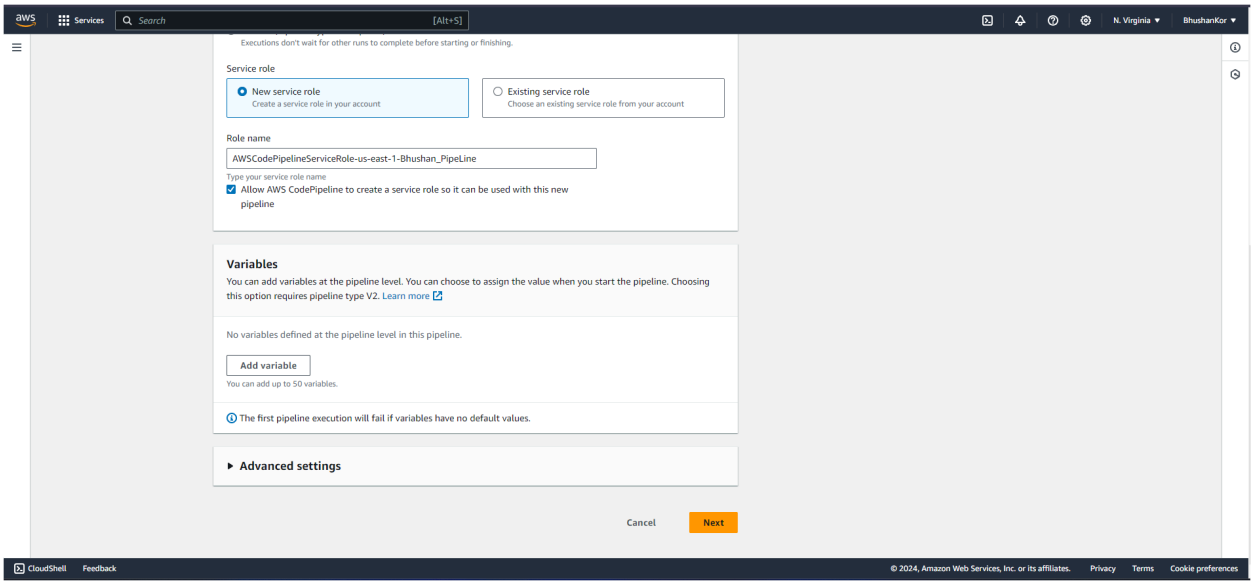


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

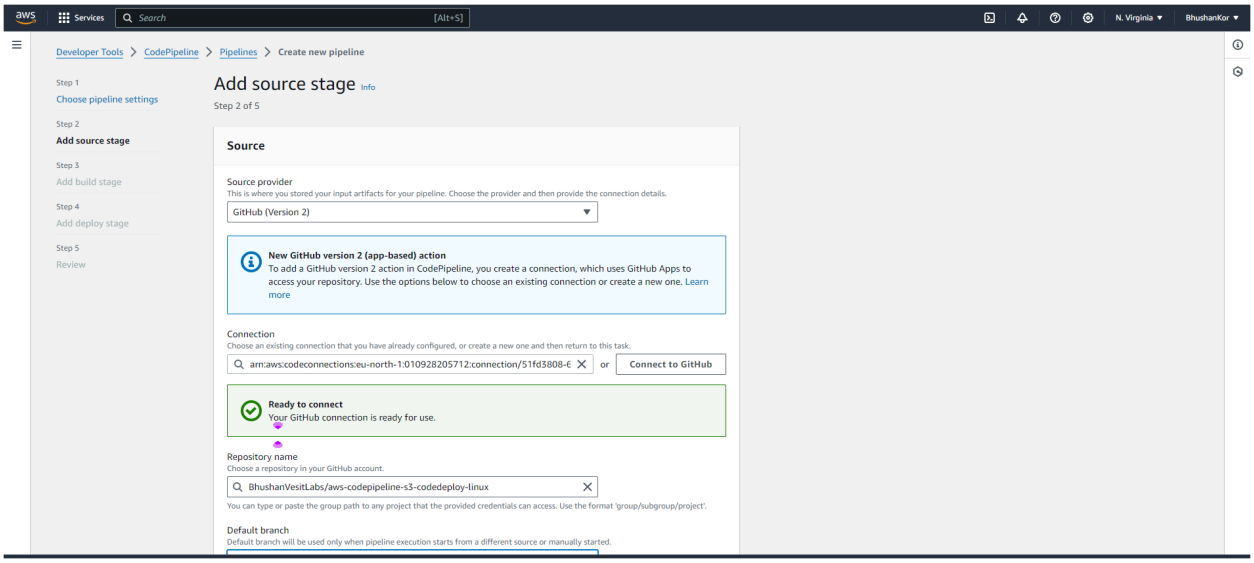
Step 2: Give the name to the pipeline, select a new service role, and keep the rest all to default.



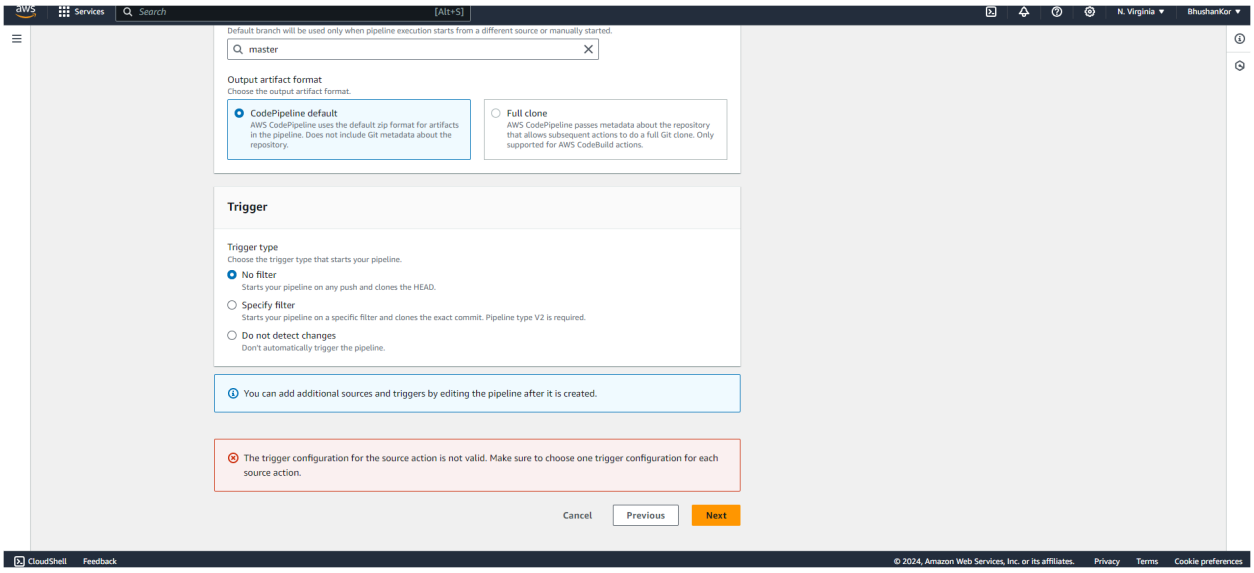


Before Step 3 Fork or clone the below repository to your GitHub account.
https://github.com/aws-samples/aws-codepipeline-s3-codedeploy-linux.git

Step 3: Now in the Add source stage Select GitHub (Version 2) [This will work only in personal AWS academy], then connect your GitHub account and select repository name, branch to main/master.



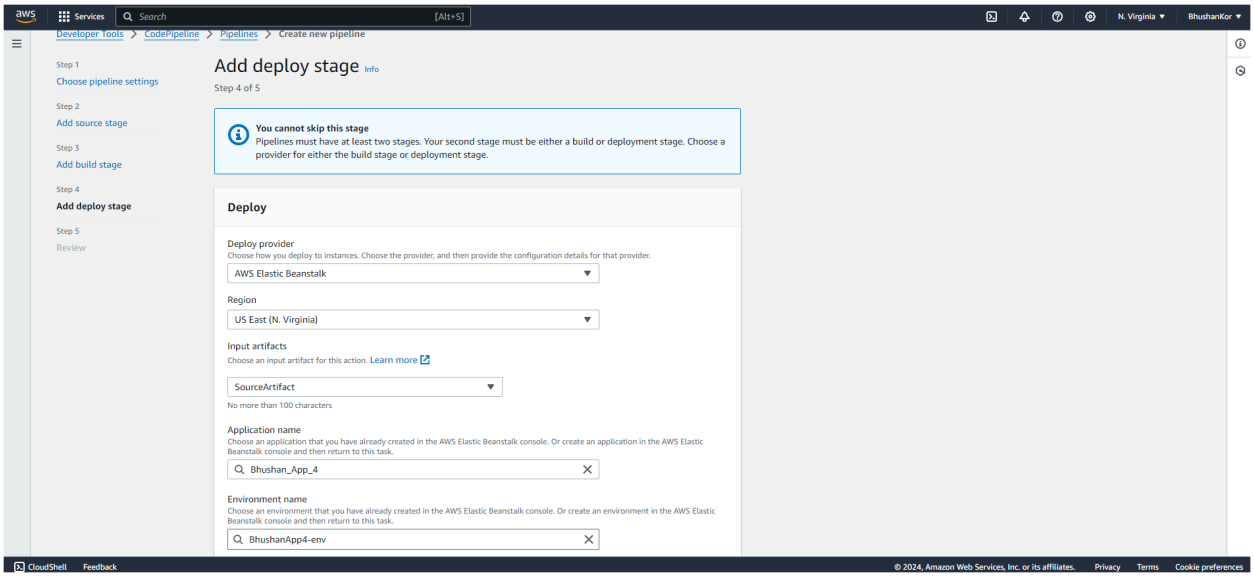
Step 4: Select Trigger to No filter otherwise it will give an error.

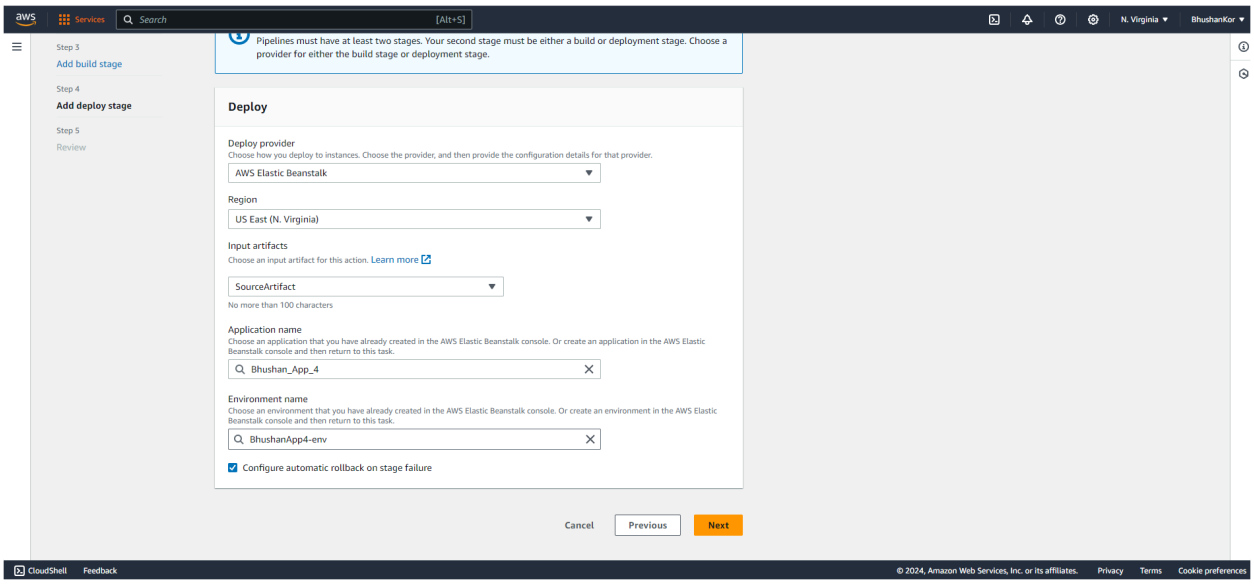


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 5: In the Depoly stage select AWS ElasticBeanstalk as the deploy provider, Select a region to US East (N.Virginia) or any with N.Virginia. Input artifacts to default and the Application name and environment name of your Elastic Beanstalk.

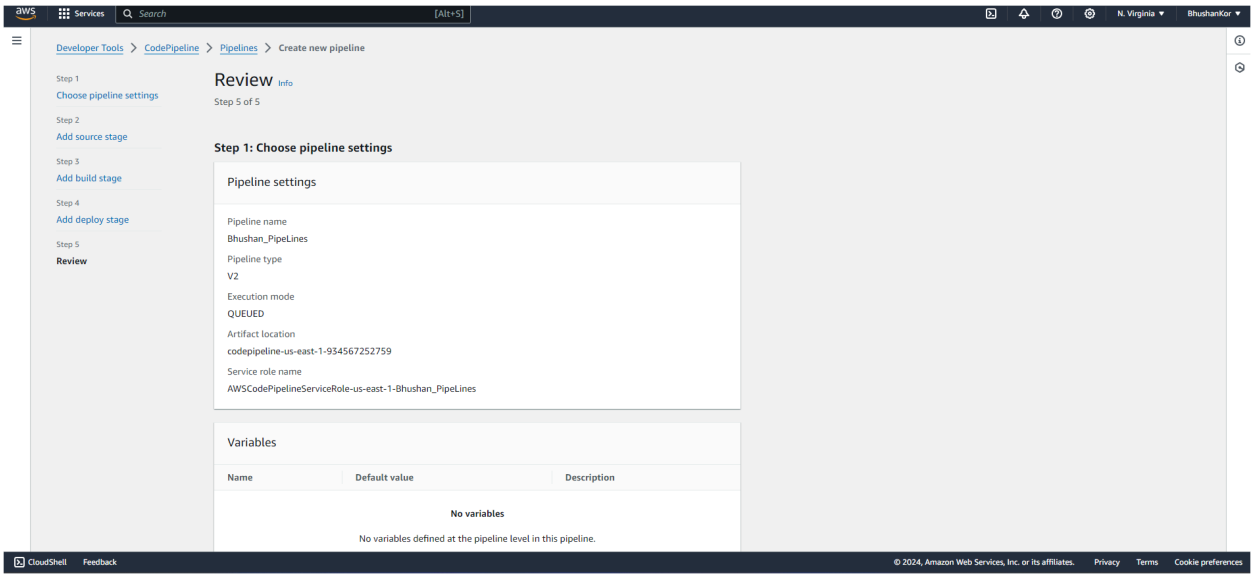


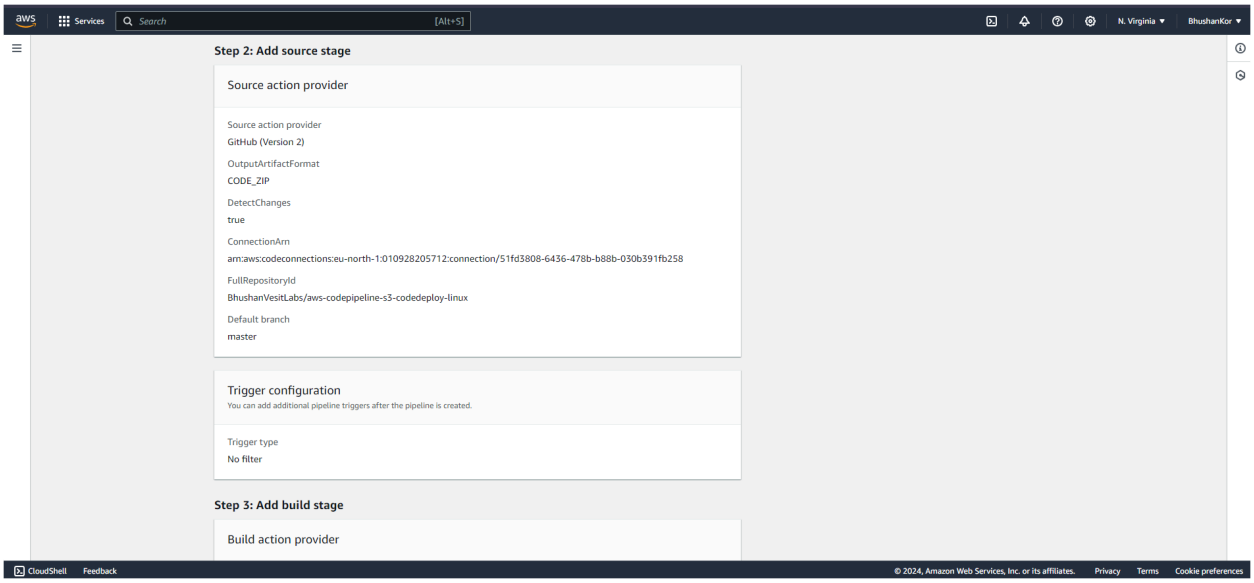


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

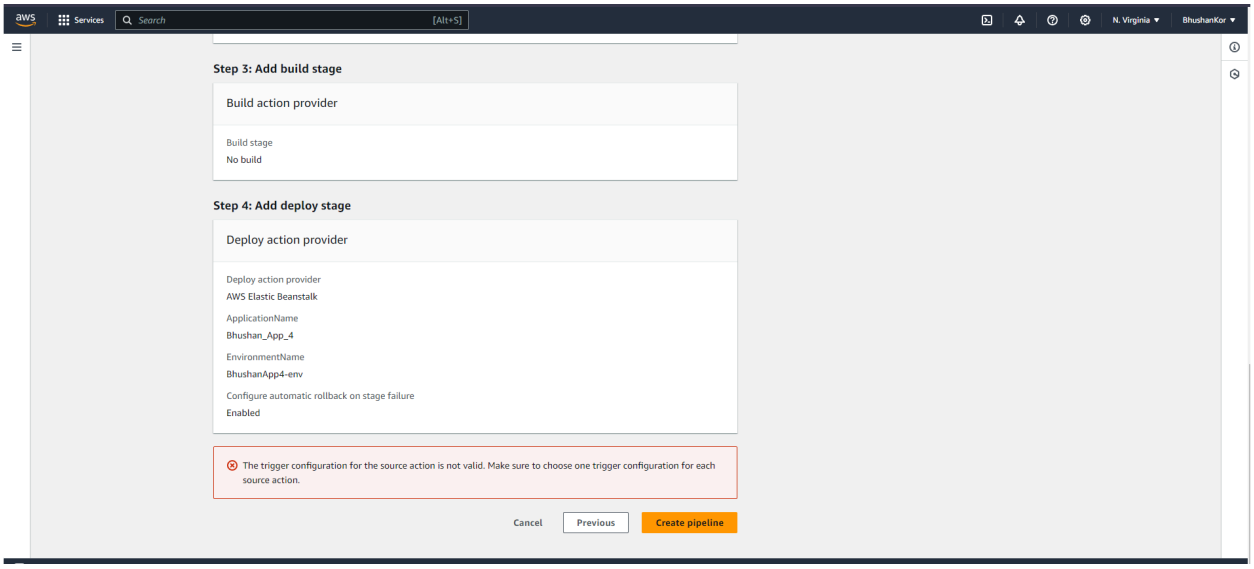
Step 6: Review and Click on Create Pipeline.



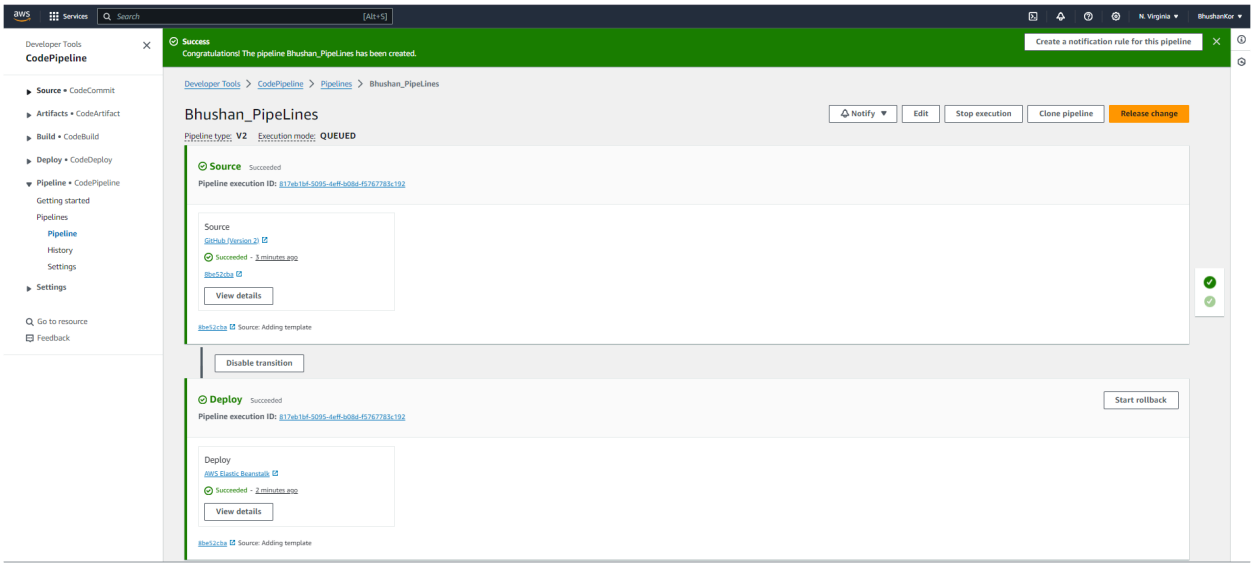


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



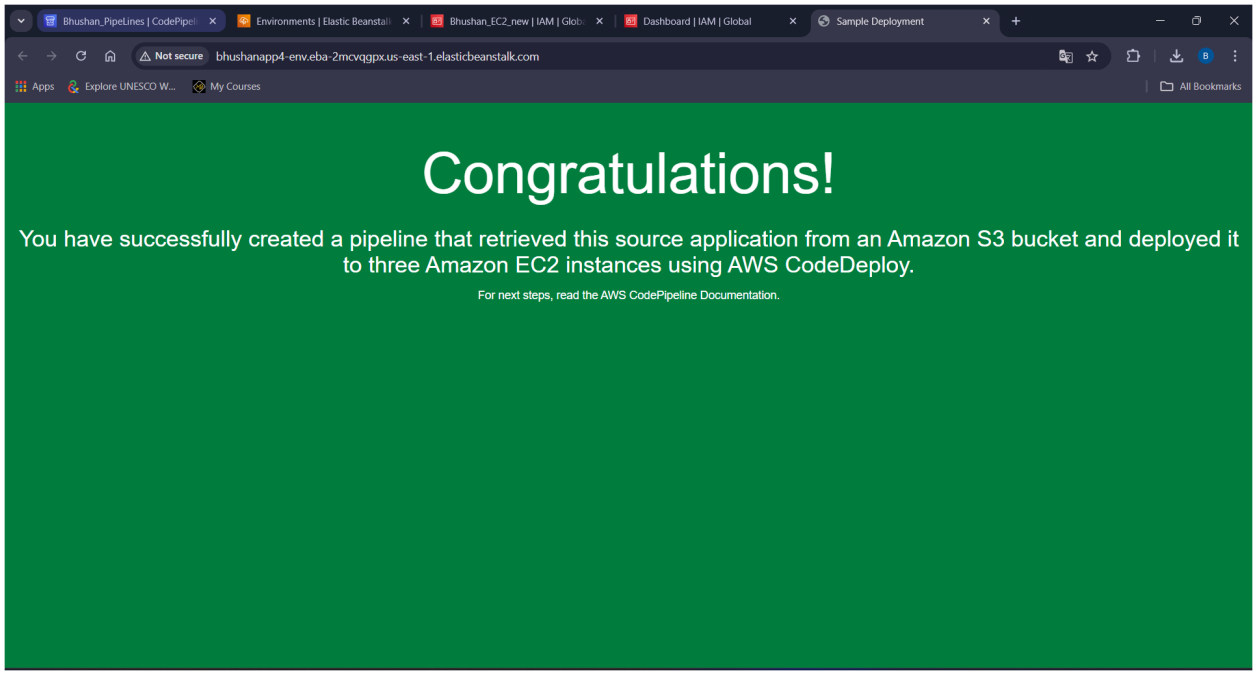
Step 7: Your Pipeline is Created Successfully.Also it is successfully connected to the source and successfully Deployed.



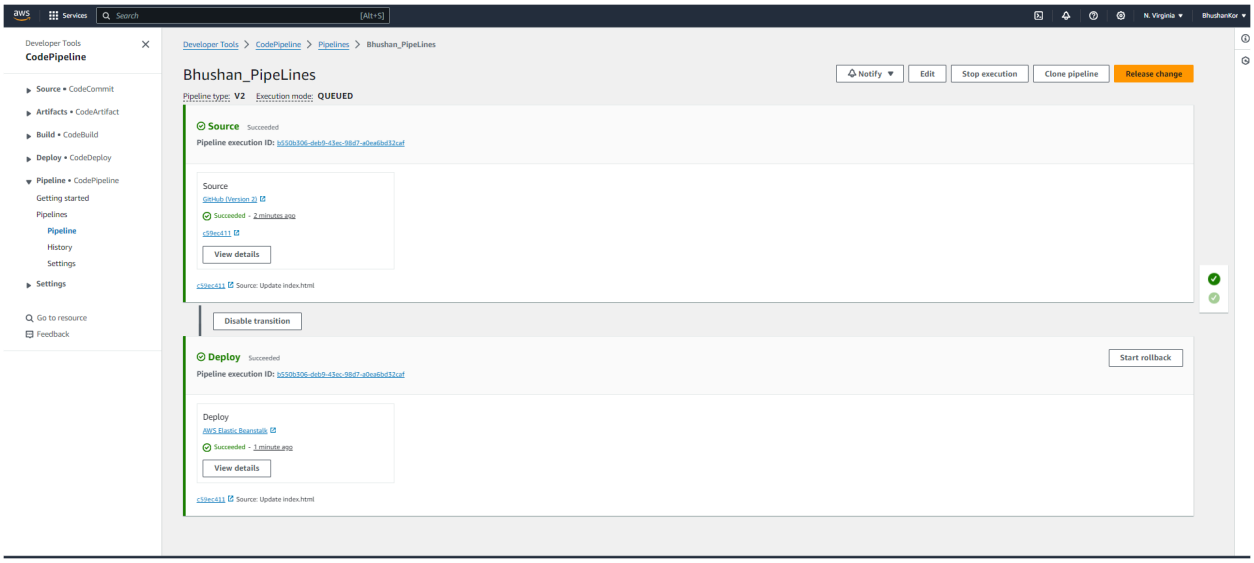
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 8: Click on the given link and you will see the Result.



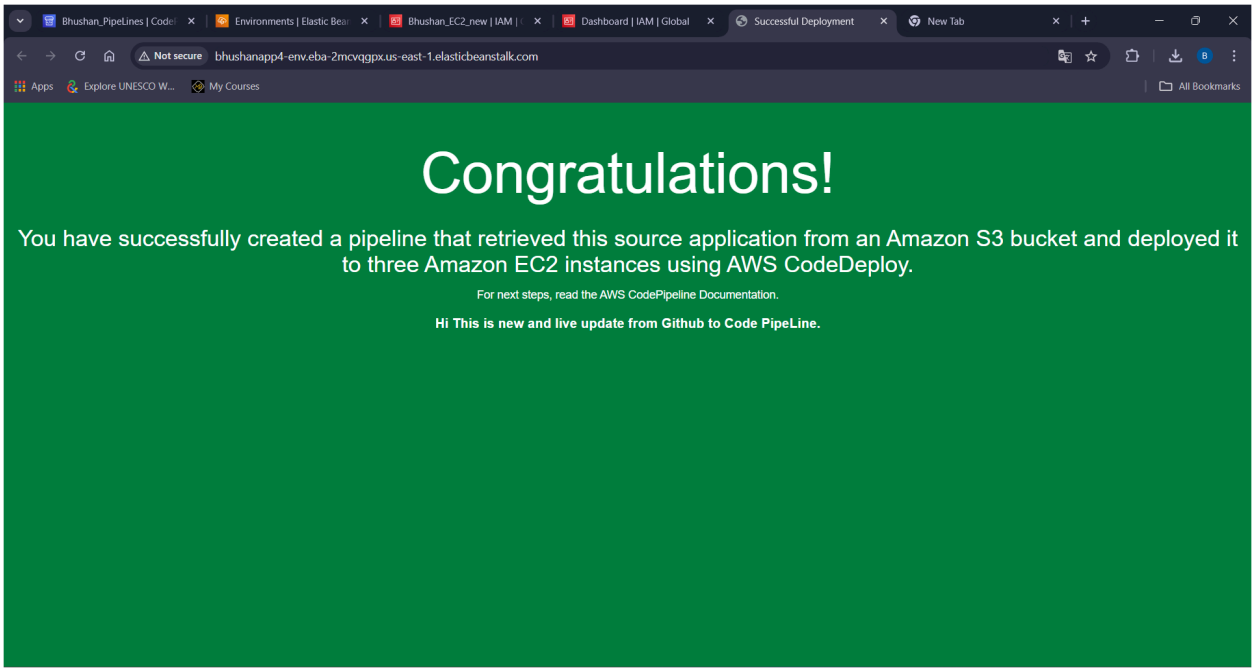
Step 9: Make Some changes in the Source code file on GitHub and commit changes and then deploy the changes in pipeline.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 10: Name of website is changed also one we line is added at the end.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

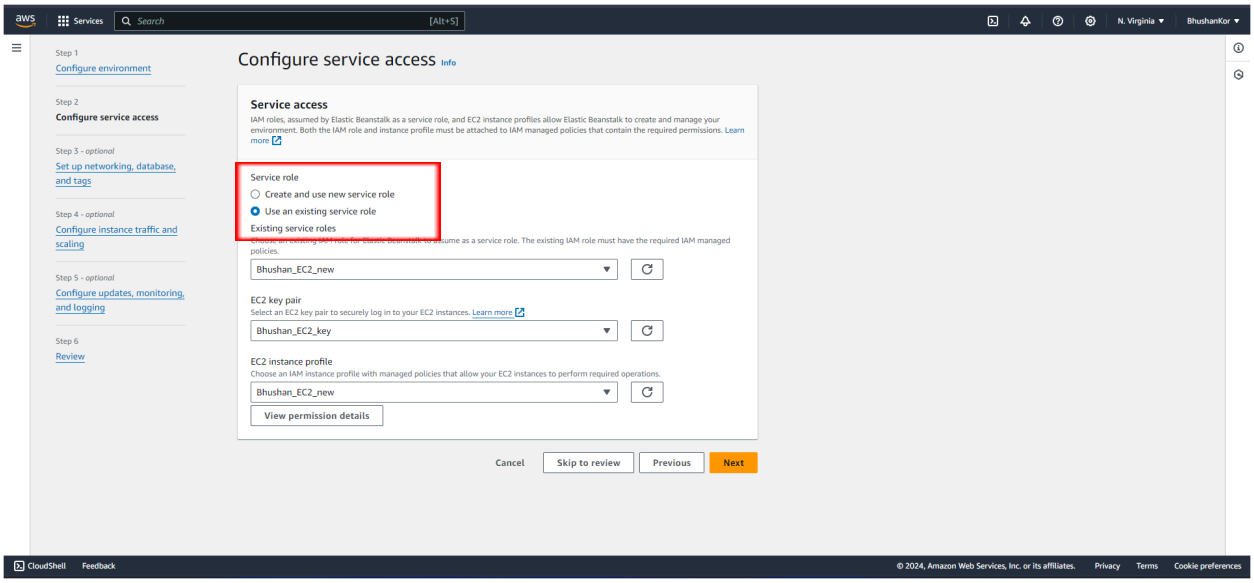
Division: D15C **Roll No: 28**

New Learnings in this experiment:

Note:

Previously Elastic Beanstalk created a default EC2 instance profile named `aws-elasticbeanstalk-ec2-role` the first time an AWS account created an environment. This instance profile included default managed policies. If your account already has this instance profile, it will remain available for you to assign to your environments.

However, recent AWS security guidelines don't allow an AWS service to automatically create roles with trust policies to other AWS services, EC2 in this case. Because of these security guidelines, Elastic Beanstalk no longer creates a default `aws-elasticbeanstalk-ec2-role` instance profile.



Because of the above problem, we have chosen the existing role and for that, we have to create a new role with certain policies.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

To create an instance profile

1. Open the Roles page in the IAM console.
2. Choose Create role.
3. Under the Trusted entity type, choose AWS service.
4. Under Use case, choose EC2.
5. Choose Next.
6. Attach the appropriate managed policies provided by Elastic Beanstalk and any additional policies that provide permissions that your application needs.
7. Choose Next.
8. Enter a name for the role.
9. (Optional) Add tags to the role.
10. Choose Create role.

For Point 6 Do the below Steps

To add managed policies to the role attached to the default instance profile

1. Type `AWSElasticBeanstalk` to filter the policies.
2. Select the following policies, and then choose Attach policy:
   - `AWSElasticBeanstalkWebTier`
   - `AWSElasticBeanstalkWorkerTier`
   - `AWSElasticBeanstalkMulticontainerDocker`
3. Also, add `AmazonS3FullAccess` or `AmazonDynamoDBFullAccess`.
4. Choose Attach policy.

Add the Trust relationship policy for EC2

To allow the EC2 instances in your environment to assume the required role, the instance profile must specify Amazon EC2 as a trusted entity in the trust relationship policy, as follows.

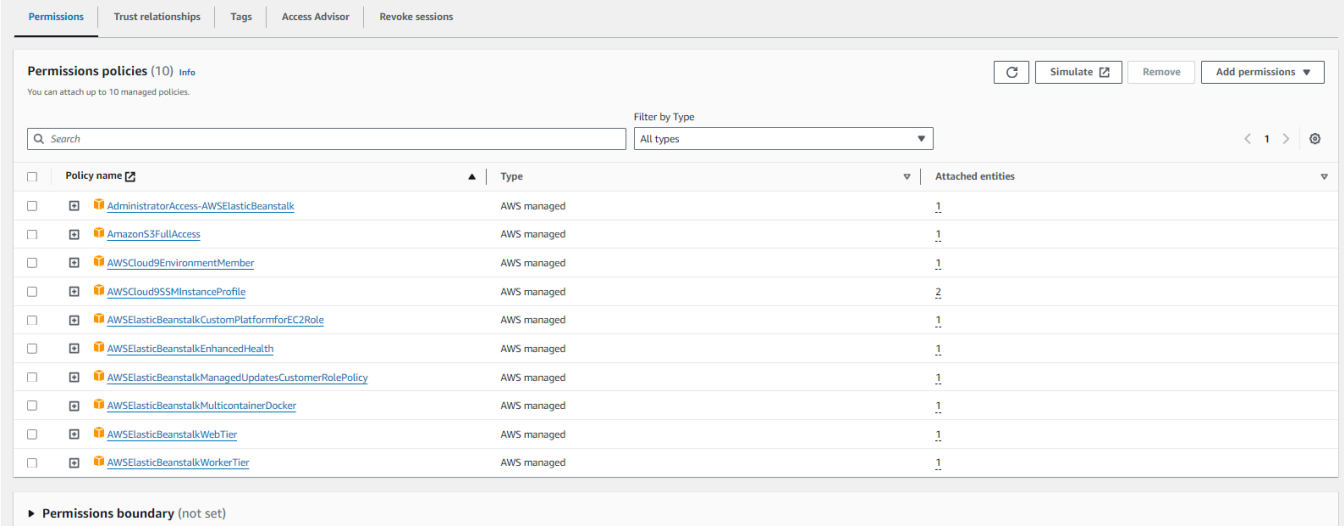
```
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
            "Service": "ec2.amazonaws.com"
        },
        "Action": "sts:AssumeRole"
    }
  ]

}
```

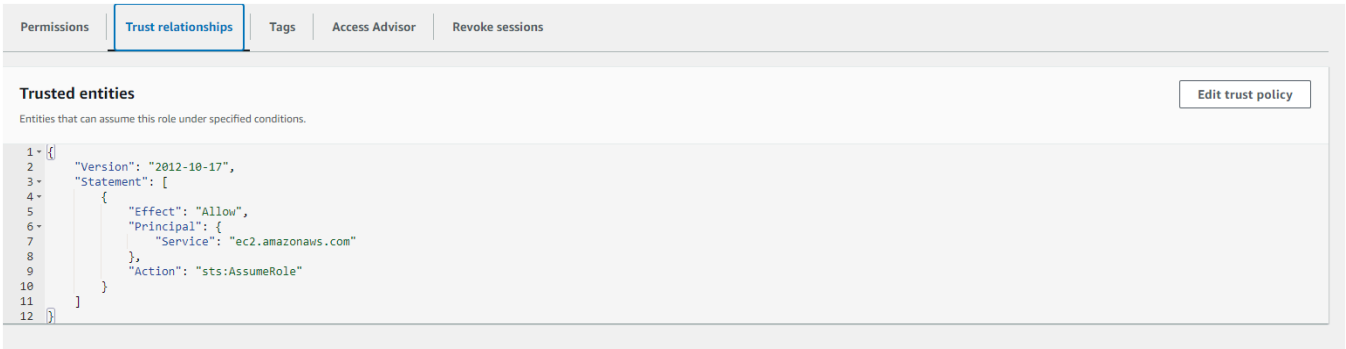
To customize permissions, you can add policies to the role attached to the default instance profile or create your own instance profile with a restricted set of permissions.

Screenshot for reference

Policies:



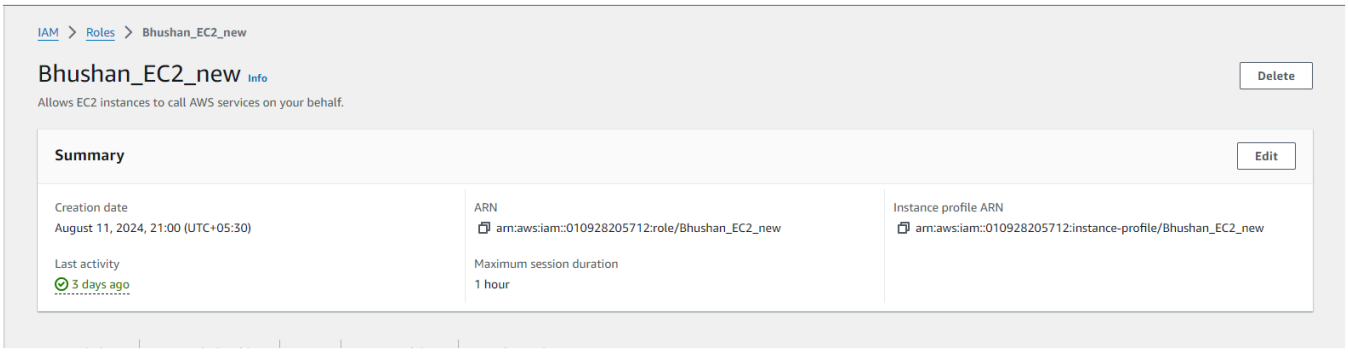
Trust Relationship:



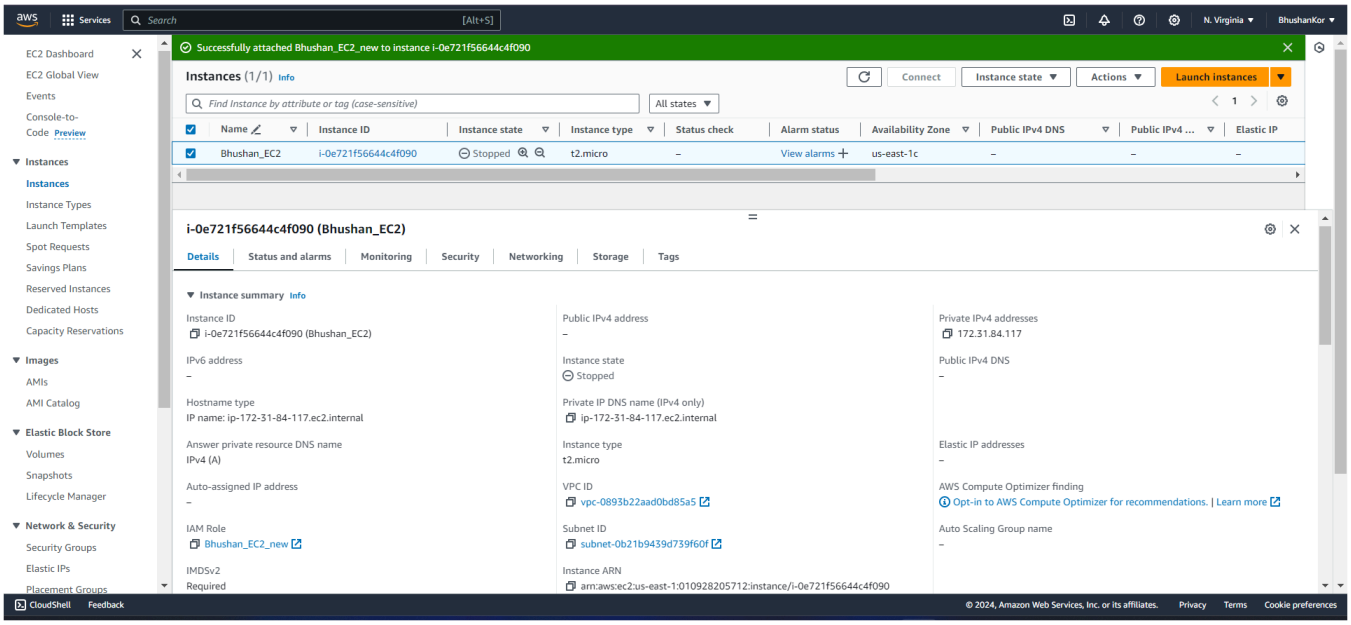
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

IAM Role:



EC2 Instance with attached IAM Role:



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Theory:

Container-based microservices architectures have revolutionized how development and operations teams test and deploy modern software. Containers allow companies to scale and deploy applications more efficiently, but they also introduce new challenges, adding complexity by creating a whole new infrastructure ecosystem.

Today, both large and small software companies are deploying thousands of container instances daily. Managing this level of complexity at scale requires advanced tools. Enter Kubernetes.

Originally developed by Google, Kubernetes is an open-source container orchestration platform designed to automate the deployment, scaling, and management of containerized applications. Kubernetes has quickly become the de facto standard for container orchestration and is the flagship project of the Cloud Native Computing Foundation (CNCF), supported by major players like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

Kubernetes simplifies the deployment and operation of applications in a microservice architecture by providing an abstraction layer over a group of hosts. This allows development teams to deploy their applications while Kubernetes takes care of key tasks, including:

- Managing resource consumption by applications or teams
- Distributing application load evenly across the infrastructure
- Automatically load balancing requests across multiple instances of an application
- Monitoring resource usage to prevent applications from exceeding resource limits and automatically restarting them if needed
- Moving application instances between hosts when resources are low or if a host fails
- Automatically utilizing additional resources when new hosts are added to the cluster
- Facilitating canary deployments and rollbacks with ease

Necessary Requirements:

● **EC2 Instance:** The experiment required launching a t2.medium EC2 instance with 2 CPUs, as Kubernetes demands sufficient resources for effective functioning.

● **Minimum Requirements:**

  ○ **Instance Type:** t2.medium
  ○ **CPUs:** 2
  ○ **Memory:** Adequate for container orchestration.

This ensured that the Kubernetes cluster had the necessary resources to function smoothly

Note:

AWS Personal Account is preferred but we can also perform it on AWS Academy(adding some ignores in the command if any error occurs in below as the below experiment is performed on Personal Account .).If You are using AWS Academy Account Errors you will face in kubeadm init command so you have to add some ignores with this command.

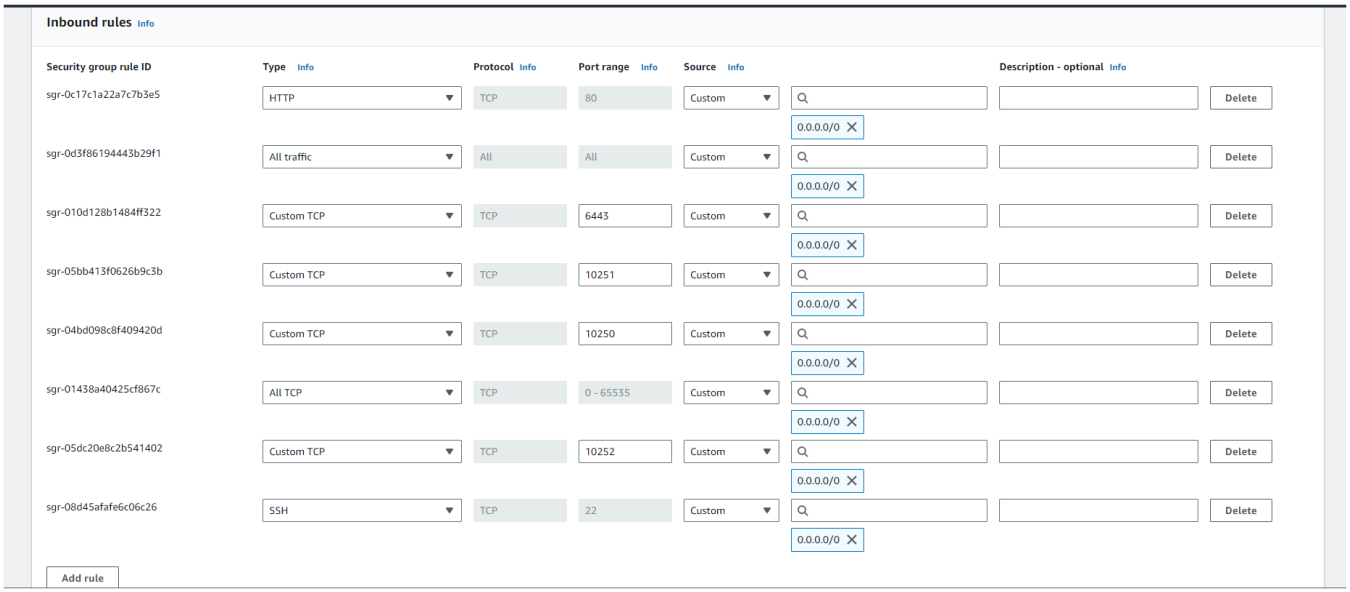
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

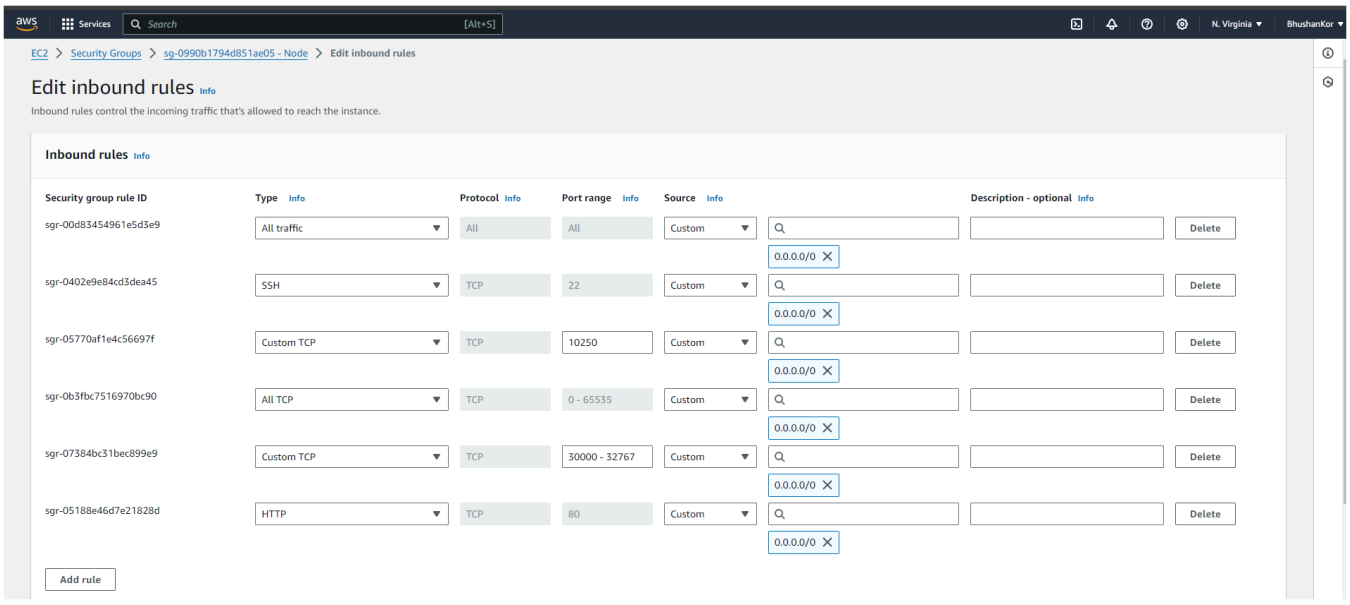
Prerequisites :

Create 2 Security Groups for Master and Nodes and add the following rules inbound rules in those Groups.

Master:



Node :

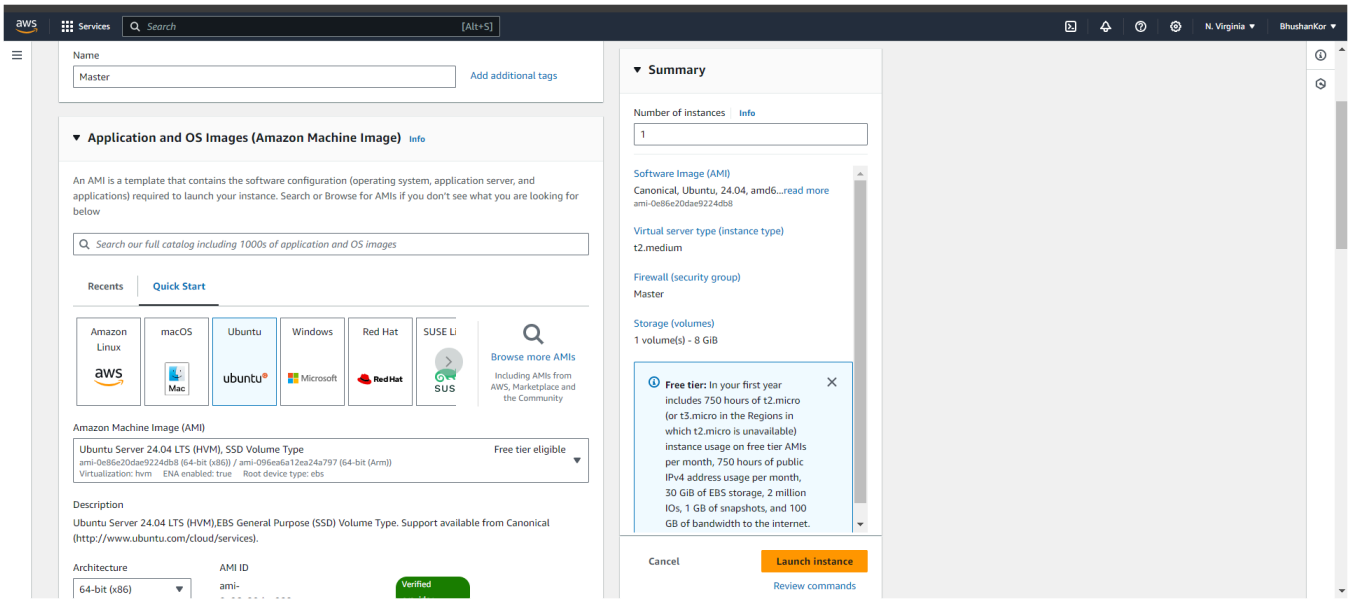


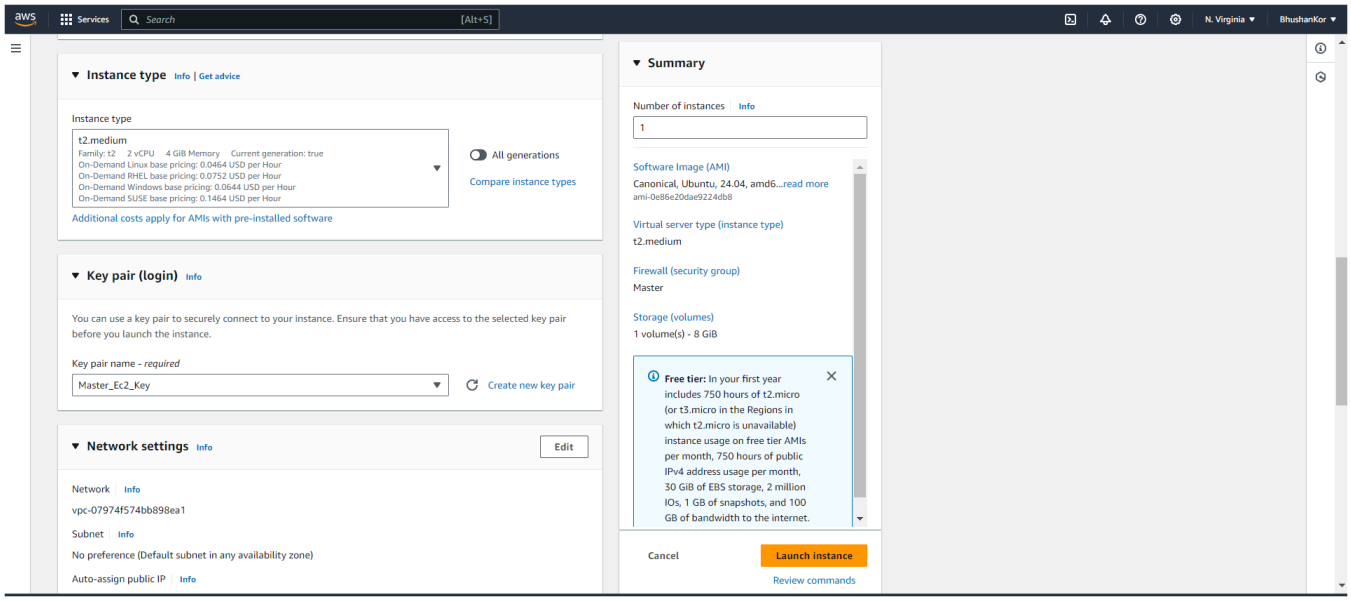
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 1: Log in to your AWS Academy/personal account and launch 3 new Ec2 Instances.
Select Ubuntu as AMI and t2.medium as Instance Type and create a key of type RSA with .pem extension and move the downloaded key to the new folder.We can use 3 Different keys or 1 common key also.
Note: A minimum of 2 CPUs are required so Please select t2.medium and do not forget to stop the instance after the experiment because it is not available in the free tier.
Also Select Security groups from existing.

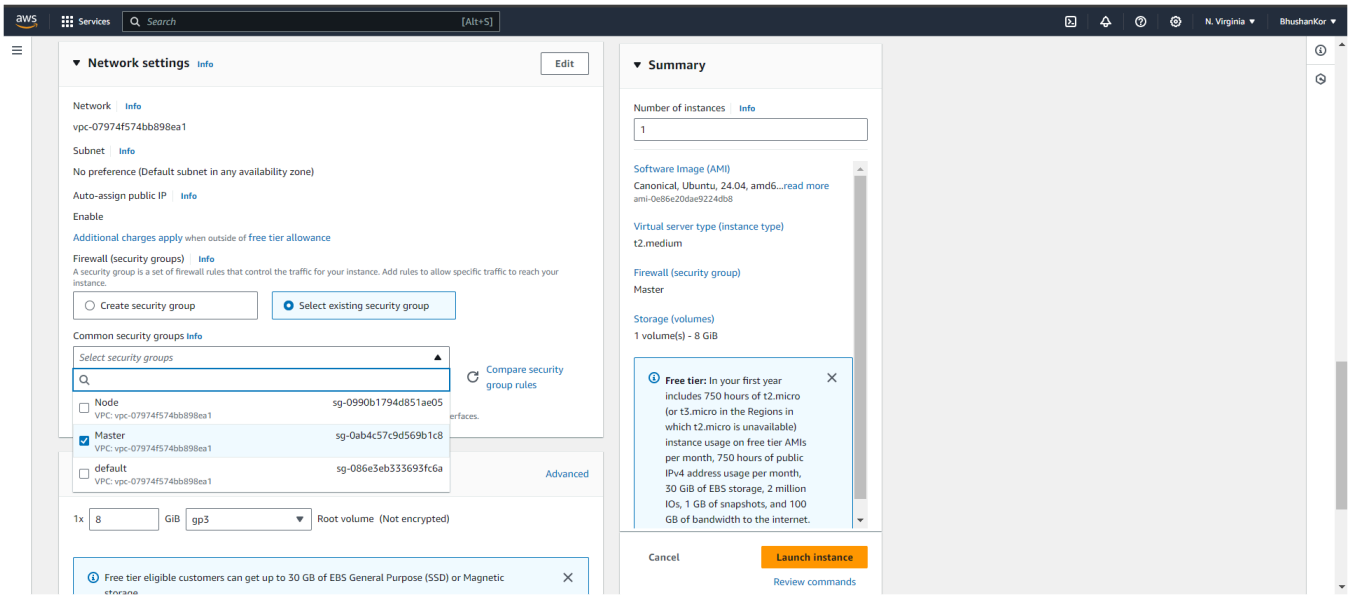
Master:





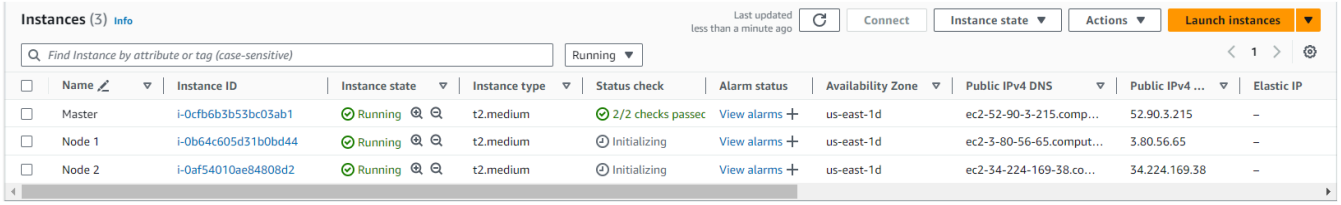
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

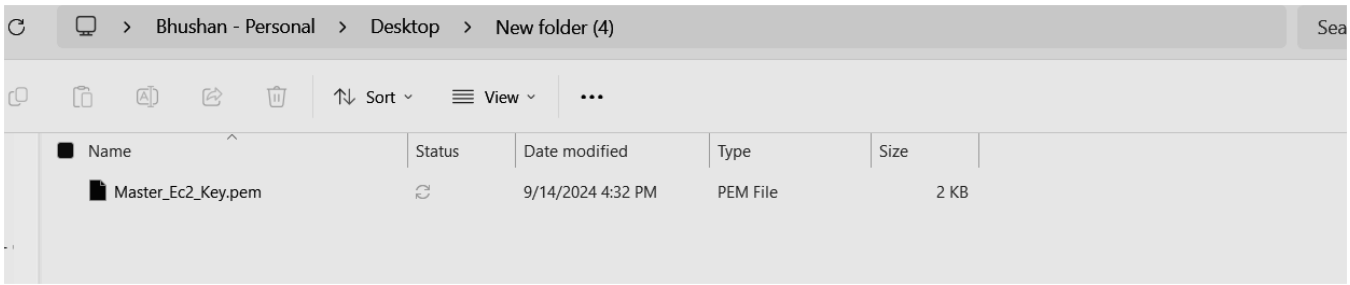


Do Same for 2 Nodes and use security groups of Node for that.

Step 2: After creating the instances click on Connect & connect all 3 instances and navigate to SSH Client.



(Downloded Key

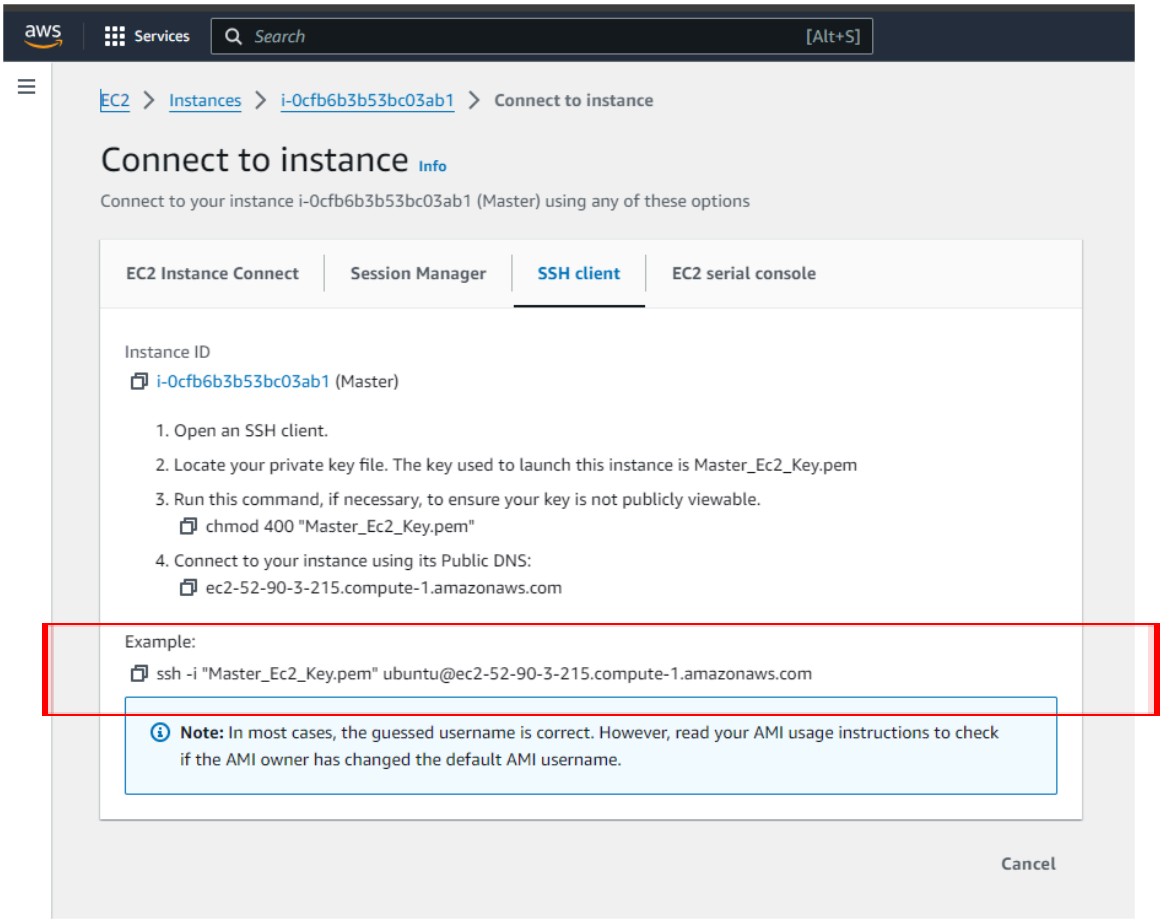


)

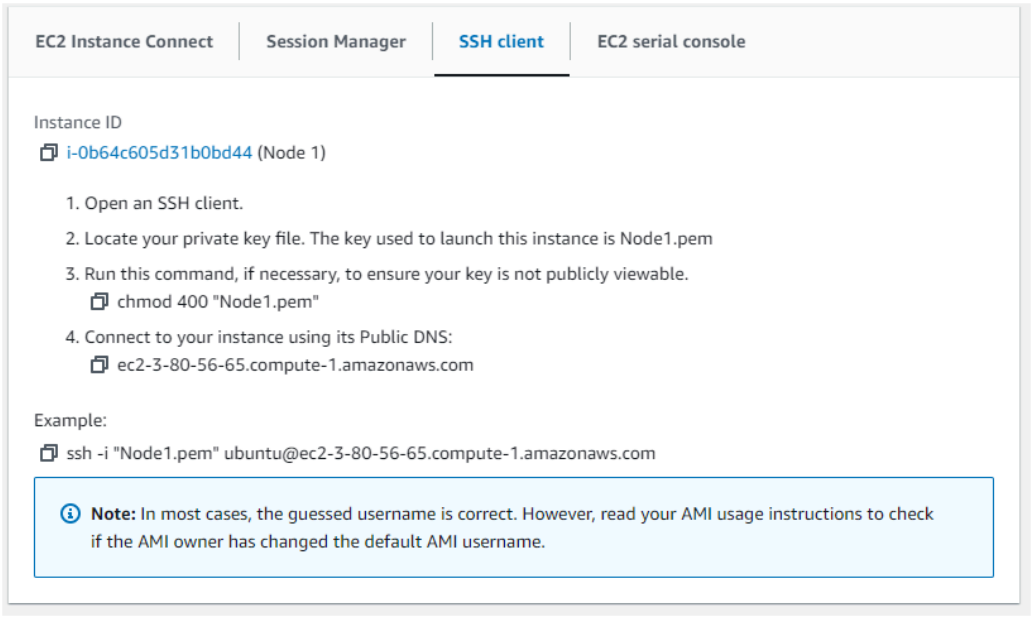
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 3: Now open the folder in the terminal 3 times for Master, Node1& Node 2 where our .pem key is stored and paste the Example command (starting with ssh -i …..) in the terminal.(ssh -i "Master_Ec2_Key.pem" ubuntu@ec2-54-196-129-215.compute-1.amazonaws.com)
Master:



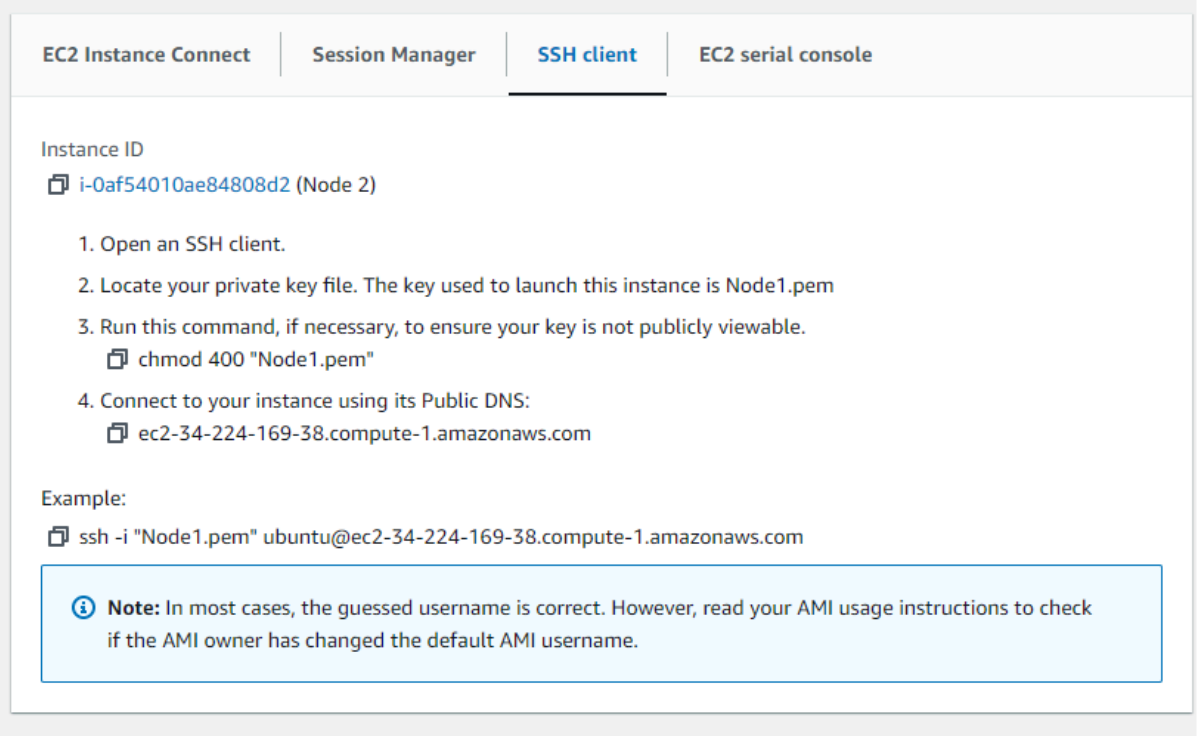
Node 1:



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

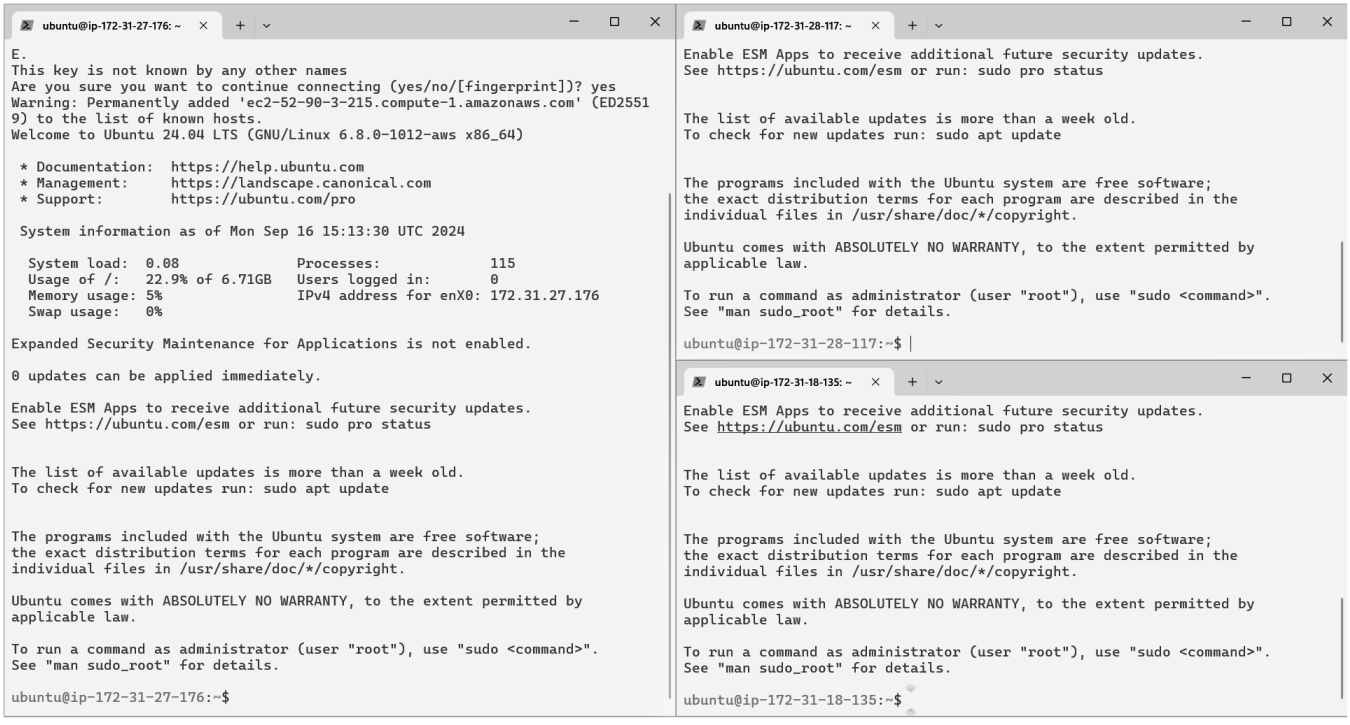
Division: D15C **Roll No: 28**

Node 2:



Here I have use 2 keys 1 for master and 1 for 2 node so I have to run open 3 terminals.In master key folder 1 terminal and 2 terminals in node 1 key folder.
If you use 1 Key only, you can open 3 terminal in one folder only.

Successful Connection:



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 4: Run on Master,Node 1,and Node 2 the below commands to install and setup Docker in Master, Node1, and Node2.

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

```
ubuntu@ip-172-31-27-176:~$
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/
trusted.gpg.d/docker.gpg > /dev/null

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/
ubuntu $(lsb_release -cs) stable"
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instea
d (see apt-key(8)).
OK
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble
 stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docke
r_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_d
ownload_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
 [126 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelea
se [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Pa
ckages [15.0 MB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [
13.8 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [
354 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [
79.4 kB]
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
cted amd64 c-n-f Metadata [116 B]
Get:51 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiv
erse amd64 Components [212 B]
Get:52 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiv
erse amd64 c-n-f Metadata [116 B]
Fetched 28.9 MB in 4s (6976 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s
) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file h
as an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is st
ored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATI
ON section in apt-key(8) for details.
ubuntu@ip-172-31-27-176:~$
```

sudo apt-get update
sudo apt-get install -y docker-ce

```
ubuntu@ip-172-31-27-176:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelea
se
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s
) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file h
as an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is st
ored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATI
ON section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 133 not upgraded.
Need to get 122 MB of archives.
After this operation, 440 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pi
gz amd64 2.8-1 [65.6 kB]
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service →
 /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /us
r/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-27-176:~$
```

sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF

```
ubuntu@ip-172-31-27-176:~$ sudo mkdir -p /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@ip-172-31-27-176:~$
```

sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker

```
ubuntu@ip-172-31-27-176:~$ sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
Synchronizing state of docker.service with SysV service script with /usr/lib
/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
ubuntu@ip-172-31-27-176:~$
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 5: Run the below command to install Kubernets.
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

```
ubuntu@ip-172-31-27-176:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.3
1/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-k
eyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://p
kgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/ku
bernetes.list

deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8
s.io/core:/stable:/v1.31/deb/ /
ubuntu@ip-172-31-27-176:~$
```

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

```
ubuntu@ip-172-31-27-176:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelea
se
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/s
table:/v1.31/deb  InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/s
table:/v1.31/deb  Packages [4865 B]
Fetched 6051 B in 1s (11.1 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s
) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file h
as an unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is st
ored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATI
ON section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@ip-172-31-27-176:~$
```

sudo systemctl enable --now kubelet
sudo apt-get install -y containerd

```
ubuntu@ip-172-31-27-176:~$ sudo systemctl enable --now kubelet
ubuntu@ip-172-31-27-176:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer requir
ed:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 133 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
```

```
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-27-176:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
```

```
[timeouts]
  "io.containerd.timeout.bolt.open" = "0s"
  "io.containerd.timeout.metrics.shimstats" = "2s"
  "io.containerd.timeout.shim.cleanup" = "5s"
  "io.containerd.timeout.shim.load" = "5s"
  "io.containerd.timeout.shim.shutdown" = "3s"
  "io.containerd.timeout.task.state" = "2s"

[ttrpc]
  address = ""
  gid = 0
  uid = 0
```

sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd

```
ubuntu@ip-172-31-27-176:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; enabled; p>
     Active: active (running) since Mon 2024-09-16 15:31:58 UTC; 210ms ago
       Docs: https://containerd.io
   Main PID: 4763 (containerd)
      Tasks: 7
     Memory: 13.9M (peak: 14.4M)
        CPU: 50ms
     CGroup: /system.slice/containerd.service
             └─4763 /usr/bin/containerd

Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 containerd[4763]: time="2024-09-16T15:31:5>
Sep 16 15:31:58 ip-172-31-27-176 systemd[1]: Started containerd.service - c>
```

sudo apt-get install -y socat

```
ubuntu@ip-172-31-27-176:~$  sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer requir
ed:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 133 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat
amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (11.2 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-27-176:~$ |
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 6: Initialize the Kubecluster .Now Perform this Command only for Master.
sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-27-176:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0916 15:39:33.685919    5313 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that
used by kubeadm.It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-27-176 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster
.local] and IPs [10.96.0.1 172.31.27.176]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-27-176 localhost] and IPs [172.31.27.176 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-27-176 localhost] and IPs [172.31.27.176 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
[control-plane] Creating static Pod manifest for "kube-scheduler"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Starting the kubelet
[wait-control-plane] Waiting for the kubelet to boot up the control plane as static Pods from directory "/etc/kubernetes/manifests"
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.001059178s
```

```
[api-check] Waiting for a healthy API server. This can take up to 4m0s
[api-check] The API server is healthy after 6.500965245s
[upload-config] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[kubelet] Creating a ConfigMap "kubelet-config" in namespace kube-system with the configuration for the kubelets in the cluster
[upload-certs] Skipping phase. Please see --upload-certs
[mark-control-plane] Marking the node ip-172-31-27-176 as control-plane by adding the labels: [node-role.kubernetes.io/control-plane node.kubernetes.io/excl
ude-from-external-load-balancers]
[mark-control-plane] Marking the node ip-172-31-27-176 as control-plane by adding the taints [node-role.kubernetes.io/control-plane:NoSchedule]
[bootstrap-token] Using token: ttay2x.n0sqeukjai8sgfg3
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.27.176:6443 --token ttay2x.n0sqeukjai8sgfg3 \
        --discovery-token-ca-cert-hash sha256:d6fc5fb7e984c83e2807780047fec6c4f2acfe9da9184ecc028d77157608fbb6
ubuntu@ip-172-31-27-176:~$
```

Run this command on master and also copy and save the Join command from above.

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
ubuntu@ip-172-31-27-176:~$ mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-27-176:~$
```

Step 7: Now Run the command kubectl get nodes to see the nodes before executing Join command on nodes.

```
Every 2.0s: kubectl get nodes                                                    ip-172-31-27-176: Mon Sep 16 15:45:34 2024

NAME              STATUS     ROLES           AGE     VERSION
ip-172-31-27-176   NotReady   control-plane   5m38s   v1.31.1
```

Step 8: Now Run the following command on Node 1 and Node 2 to Join to master.

**sudo kubeadm join 172.31.27.176:6443 --token ttay2x.n0sqeukjai8sgfg3 **
--discovery-token-ca-cert-hash
sha256:d6fc5fb7e984c83e2807780047fec6c4f2acfe9da9184ecc028d77157608fbb6

Node 1:

```
ubuntu@ip-172-31-28-117:~$ sudo kubeadm join 172.31.27.176:6443 --token ttay2x.n0sqeukjai8sgfg3 \
        --discovery-token-ca-cert-hash sha256:d6fc5fb7e984c83e2807780047fec6c4f2acfe9da9184ecc028d77157608fbb6

[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 501.396793ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

ubuntu@ip-172-31-28-117:~$
```

Node 2:

```
ubuntu@ip-172-31-18-135:~$ sudo kubeadm join 172.31.27.176:6443 --token ttay2x.n0sqeukjai8sgfg3 \
        --discovery-token-ca-cert-hash sha256:d6fc5fb7e984c83e2807780047fec6c4f2acfe9da9184ecc028d77157608fbb6
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 1.001003808s
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

ubuntu@ip-172-31-18-135:~$
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 9: Now Run the command kubectl get nodes to see the nodes after executing Join command on nodes.

```
ubuntu@ip-172-31-27-176:~$ kubectl get nodes
NAME                STATUS     ROLES           AGE     VERSION
ip-172-31-18-135    NotReady   <none>          88s     v1.31.1
ip-172-31-27-176    NotReady   control-plane   10m     v1.31.1
ip-172-31-28-117    NotReady   <none>          2m58s   v1.31.1
```

Step 10: Since Status is NotReady we have to add a network plugin. And also we have to give the name to the nodes.

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

```
ubuntu@ip-172-31-27-176:~$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
poddisruptionbudget.policy/calico-kube-controllers created
serviceaccount/calico-kube-controllers created
serviceaccount/calico-node created
configmap/calico-config created
customresourcedefinition.apiextensions.k8s.io/bgpconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/bgppeers.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/blockaffinities.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/caliconodestatuses.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/clusterinformations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/felixconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/globalnetworksets.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/hostendpoints.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamblocks.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamconfigs.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipamhandles.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ippools.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/ipreservations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/kubecontrollersconfigurations.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networkpolicies.crd.projectcalico.org created
customresourcedefinition.apiextensions.k8s.io/networksets.crd.projectcalico.org created
clusterrole.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrole.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-kube-controllers created
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
```

sudo systemctl status kubelet

```
ubuntu@ip-172-31-27-176:~$ sudo systemctl status kubelet
● kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: active (running) since Mon 2024-09-16 15:40:01 UTC; 11min ago
       Docs: https://kubernetes.io/docs/
   Main PID: 5989 (kubelet)
      Tasks: 10 (limit: 4676)
     Memory: 32.6M (peak: 33.2M)
        CPU: 10.705s
     CGroup: /system.slice/kubelet.service
             └─5989 /usr/bin/kubelet --bootstrap-kubeconfig=/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=/etc/kubernetes/kubelet.conf --config=/var/>

Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497458    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497516    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497569    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497620    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497669    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:29 ip-172-31-27-176 kubelet[5989]: I0916 15:51:29.497719    5989 reconciler_common.go:245] "operationExecutor.VerifyControllerAttachedVolume s>
Sep 16 15:51:31 ip-172-31-27-176 kubelet[5989]: E0916 15:51:31.605091    5989 kubelet.go:2902] "Container runtime network not ready" networkReady="NetworkR>
Sep 16 15:51:32 ip-172-31-27-176 kubelet[5989]: I0916 15:51:32.366237    5989 scope.go:117] "RemoveContainer" containerID="f44f06967c5b3e567e07841a7b4352ae>
Sep 16 15:51:36 ip-172-31-27-176 kubelet[5989]: E0916 15:51:36.606675    5989 kubelet.go:2902] "Container runtime network not ready" networkReady="NetworkR>
Sep 16 15:51:41 ip-172-31-27-176 kubelet[5989]: E0916 15:51:41.608404    5989 kubelet.go:2902] "Container runtime network not ready" networkReady="NetworkR>
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Now Run command kubectl get nodes -o wide we can see Status is ready.

```
ubuntu@ip-172-31-27-176:~$ ubuntu@ip-172-31-27-176:~$ kubectl get nodes -o wide
NAME              STATUS   ROLES           AGE     VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-18-135  Ready    <none>          6m19s   v1.31.1   172.31.18.135   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-27-176  Ready    control-plane   15m     v1.31.1   172.31.27.176   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-28-117  Ready    <none>          7m49s   v1.31.1   172.31.28.117   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
```

Now to Rename run this command

kubectl label node ip-172-31-18-135 kubernetes.io/role=worker

Rename to Node 1:kubectl label node ip-172-31-28-117 kubernetes.io/role=Node1
Rename to Node 2:kubectl label node ip-172-31-18-135 kubernetes.io/role=Node2

```
ubuntu@ip-172-31-27-176:~$ kubectl label node ip-172-31-28-117 kubernetes.io/role=Node1
node/ip-172-31-28-117 labeled
ubuntu@ip-172-31-27-176:~$ kubectl label node ip-172-31-18-135 kubernetes.io/role=Node2
node/ip-172-31-18-135 labeled
```

Step 11: Run command kubectl get nodes -o wide . And Hence we can see we have Successfully connected Node 1 and Node 2 to the Master.

```
ubuntu@ip-172-31-27-176:~$ kubectl get nodes -o wide
NAME              STATUS   ROLES           AGE   VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE           KERNEL-VERSION   CONTAINER-RUNTIME
ip-172-31-18-135  Ready    Node2           12m   v1.31.1   172.31.18.135   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-27-176  Ready    control-plane   21m   v1.31.1   172.31.27.176   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ip-172-31-28-117  Ready    Node1           13m   v1.31.1   172.31.28.117   <none>        Ubuntu 24.04 LTS   6.8.0-1012-aws   containerd://1.7.12
ubuntu@ip-172-31-27-176:~$
```

Or run kubectl get nodes

```
ubuntu@ip-172-31-27-176:~$ kubectl get nodes
NAME              STATUS   ROLES           AGE   VERSION
ip-172-31-18-135  Ready    Node2           24m   v1.31.1
ip-172-31-27-176  Ready    control-plane   33m   v1.31.1
ip-172-31-28-117  Ready    Node1           25m   v1.31.1
ubuntu@ip-172-31-27-176:~$
```

Conclusion: In this experiment, we successfully set up a Kubernetes cluster with one master and two worker nodes on AWS EC2 instances. After installing Docker, Kubernetes tools (kubelet, kubeadm, kubectl), and containerd on all nodes, the master node was initialized and the worker nodes were joined to the cluster. Initially, the nodes were in the `NotReady` state, which was resolved by installing the Calico network plugin. We also labeled the nodes with appropriate roles (control-plane and worker). The cluster became fully functional with all nodes in the `Ready` state, demonstrating the successful configuration and orchestration of Kubernetes.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To install Kubectl and execute Kubectl commands to manage the Kubernetes cluster and deploy Your First Kubernetes Application.

Theory:

Kubernetes, originally developed by Google, is an open-source container orchestration platform. It automates the deployment, scaling, and management of containerized applications, ensuring high availability and fault tolerance. Kubernetes is now the industry standard for container orchestration and is governed by the **Cloud Native Computing Foundation (CNCF)**, with contributions from major cloud and software providers like Google, AWS, Microsoft, IBM, Intel, Cisco, and Red Hat.

Kubernetes Deployment: Is a resource in Kubernetes that provides declarative updates for Pods and ReplicaSets. With a Deployment, you can define how many replicas of a pod should run, roll out new versions of an application, and roll back to previous versions if necessary. It ensures that the desired number of pod replicas are running at all times.

Necessary Requirements:

- **EC2 Instance:** The experiment required launching a t2.medium EC2 instance with 2 CPUs, as Kubernetes demands sufficient resources for effective functioning.
- **Minimum Requirements:**
    - **Instance Type:** t2.medium
    - **CPUs:** 2
    - **Memory:** Adequate for container orchestration.

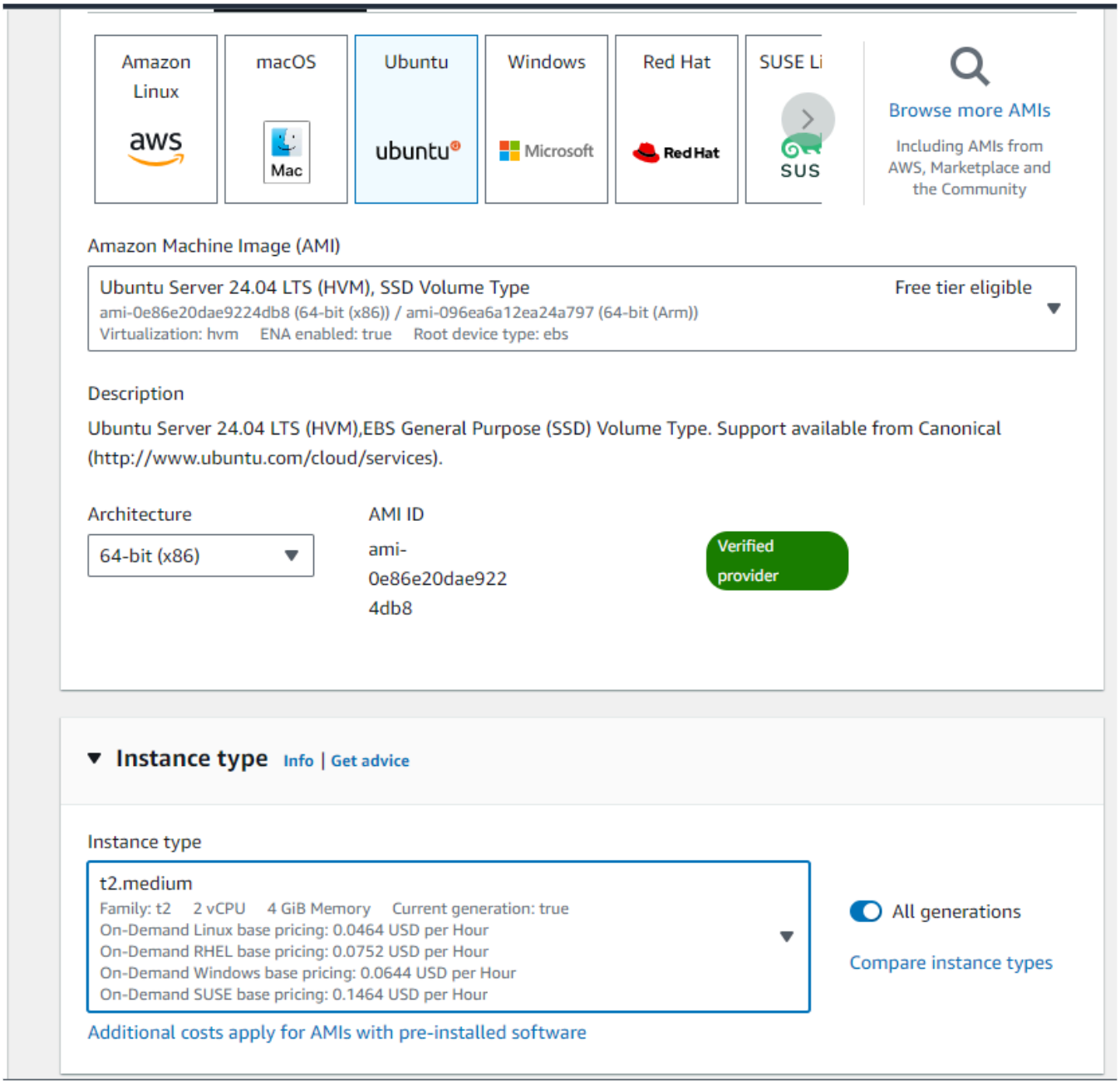
This ensured that the Kubernetes cluster had the necessary resources to function smoothly.

Note:
AWS Personal Account is preferred but we can also perform it on AWS Academy(adding some ignores in the command if any error occurs in below as the below experiment is performed on Personal Account .).
If You are using AWS Academy Account Errors you will face in kubeadm init command so you have to add some ignores with this command.

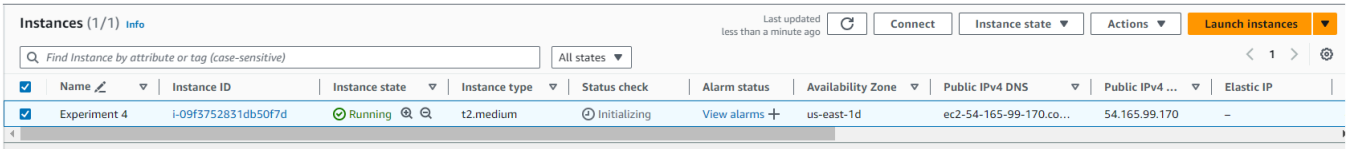
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 1: Log in to your AWS Academy/personal account and launch a new Ec2 Instance.
Select Ubuntu as AMI and t2.medium as Instance Type, create a key of type RSA with .pem extension, and move the downloaded key to the new folder.
Note: A minimum of 2 CPUs are required so Please select t2.medium and do not forget to stop the instance after the experiment because it is not available in the free tier.

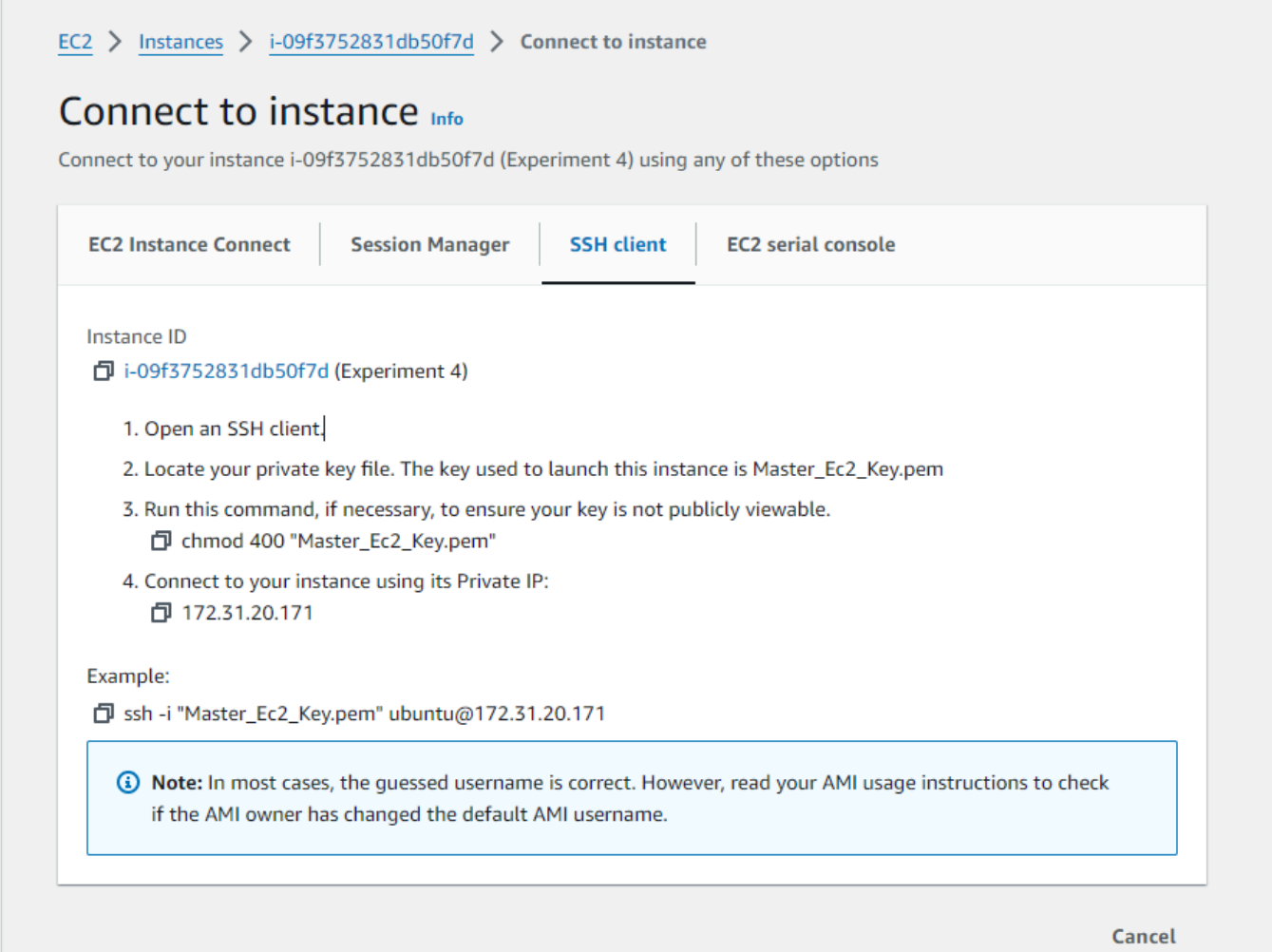


Step 2: After creating the instance click on Connect the instance and navigate to SSH Client.

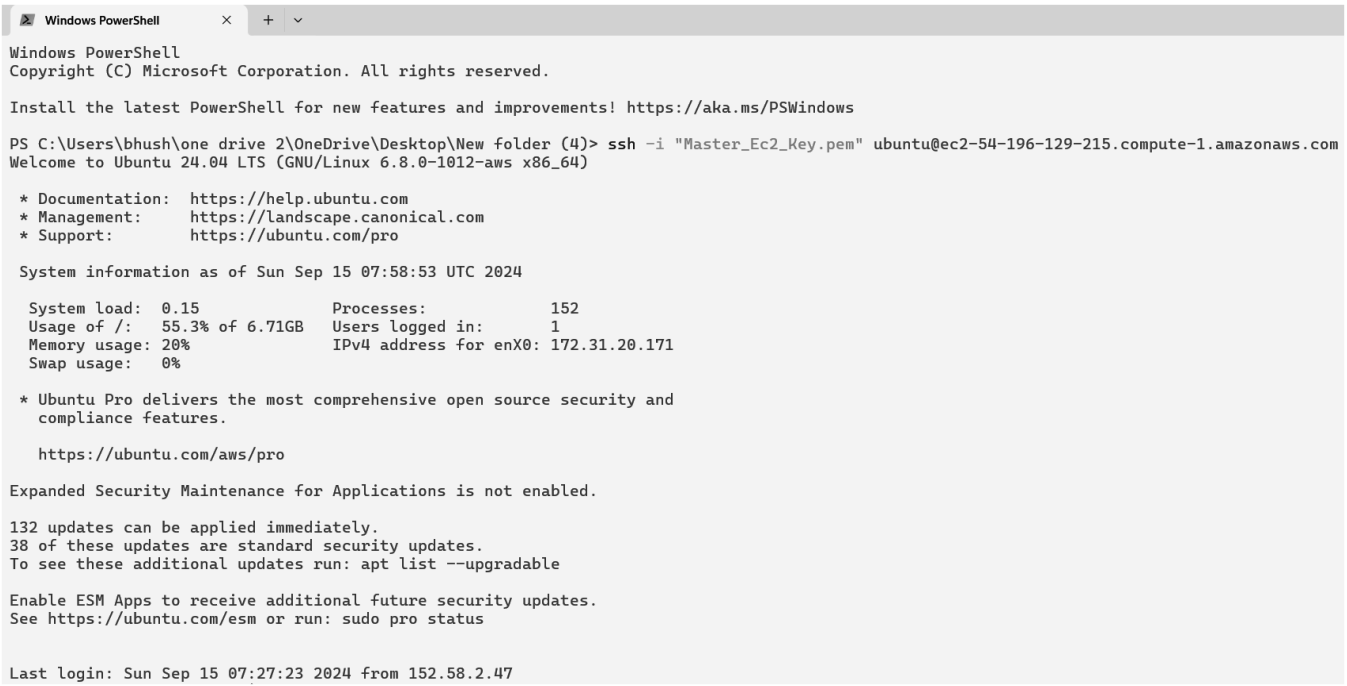


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



Step 3: Now open the folder in the terminal where our .pem key is stored and paste the Example command (starting with ssh -i …..) in the terminal.(ssh -i "Master_Ec2_Key.pem" ubuntu@ec2-54-196-129-215.compute-1.amazonaws.com)



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 4: Run the below commands to install and setup Docker.

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

```
ubuntu@ip-172-31-20-171:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo tee /etc/apt/trusted.gpg.d/docker.gpg > /dev/null
ubuntu@ip-172-31-20-171:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Found existing deb entry in /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Found existing deb-src entry in /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [13.8 kB]
Fetched 62.6 kB in 0s (128 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has a
n unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
```

sudo apt-get update
sudo apt-get install -y docker-ce

```
ubuntu@ip-172-31-20-171:~$ sudo apt-get update
sudo apt-get install -y docker-ce
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: The key(s) in the keyring /etc/apt/trusted.gpg.d/docker.gpg are ignored as the file has a
n unsupported filetype.
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION s
ection in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin libltdl7 libslirp0 pigz
  slirp4netns
0 upgraded, 10 newly installed, 0 to remove and 133 not upgraded.
Need to get 122 MB of archives.
After this operation, 440 MB of additional disk space will be used.
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 libltdl7 amd64 2.4.7-7build1 [40.3 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:4 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.7.22-1 [29.5 MB]
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-buildx-plugin amd64 0.16.2-1~ubuntu.24.04~noble [29.9 MB]
Get:7 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:27.2.1-1~ubuntu.24.04~noble [15.0 MB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:27.2.1-1~ubuntu.24.04~noble [25.6 MB]
Get:9 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-rootless-extras amd64 5:27.2.1-1~ubuntu.24.04~noble [9571 kB]
Get:10 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-compose-plugin amd64 2.29.2-1~ubuntu.24.04~noble [12.5 MB]
Fetched 122 MB in 2s (71.3 MB/s)
Selecting previously unselected package pigz.
(Reading database ... 67741 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package containerd.io.
Preparing to unpack .../1-containerd.io_1.7.22-1_amd64.deb ...
Unpacking containerd.io (1.7.22-1) ...
Selecting previously unselected package docker-buildx-plugin.
Preparing to unpack .../2-docker-buildx-plugin_0.16.2-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-buildx-plugin (0.16.2-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce-cli.
Preparing to unpack .../3-docker-ce-cli_5%3a27.2.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce-cli (5:27.2.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce.
Preparing to unpack .../4-docker-ce_5%3a27.2.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce (5:27.2.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce-rootless-extras.
Preparing to unpack .../5-docker-ce-rootless-extras_5%3a27.2.1-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce-rootless-extras (5:27.2.1-1~ubuntu.24.04~noble) ...
Selecting previously unselected package docker-compose-plugin.
Preparing to unpack .../6-docker-compose-plugin_2.29.2-1~ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-compose-plugin (2.29.2-1~ubuntu.24.04~noble) ...
Selecting previously unselected package libltdl7:amd64.
Preparing to unpack .../7-libltdl7_2.4.7-7build1_amd64.deb ...
Unpacking libltdl7:amd64 (2.4.7-7build1) ...
Selecting previously unselected package libslirp0:amd64.
Preparing to unpack .../8-libslirp0_4.7.0-1ubuntu3_amd64.deb ...
Unpacking libslirp0:amd64 (4.7.0-1ubuntu3) ...
Selecting previously unselected package slirp4netns.
Preparing to unpack .../9-slirp4netns_1.2.1-1build2_amd64.deb ...
Unpacking slirp4netns (1.2.1-1build2) ...
Setting up docker-buildx-plugin (0.16.2-1~ubuntu.24.04~noble) ...
Setting up containerd.io (1.7.22-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
```

```
Unpacking slirp4netns (1.2.1-1build2) ...
Setting up docker-buildx-plugin (0.16.2-1~ubuntu.24.04~noble) ...
Setting up containerd.io (1.7.22-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up docker-compose-plugin (2.29.2-1~ubuntu.24.04~noble) ...
Setting up libltdl7:amd64 (2.4.7-7build1) ...
Setting up docker-ce-cli (5:27.2.1-1~ubuntu.24.04~noble) ...
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up docker-ce-rootless-extras (5:27.2.1-1~ubuntu.24.04~noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:27.2.1-1~ubuntu.24.04~noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

sudo mkdir -p /etc/docker

cat <<EOF | sudo tee /etc/docker/daemon.json

{

"exec-opts": ["native.cgroupdriver=systemd"]

}

EOF

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
ubuntu@ip-172-31-20-171:~$ sudo mkdir -p /etc/docker
ubuntu@ip-172-31-20-171:~$ cat <<EOF | sudo tee /etc/docker/daemon.json
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
{
    "exec-opts": ["native.cgroupdriver=systemd"]
}
```

sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker

```
ubuntu@ip-172-31-20-171:~$ sudo systemctl enable docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
```

Step 5: Run the below command to install Kubernets.
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

```
ubuntu@ip-172-31-20-171:~$ curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyri
ng.gpg
ubuntu@ip-172-31-20-171:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /et
c/apt/sources.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

```
ubuntu@ip-172-31-20-171:~$ sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu noble InRelease
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Get:7 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  Packages [4865 B]
Fetched 6051 B in 0s (12.9 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 130 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubeadm 1.31.1-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubectl 1.31.1-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubelet 1.31.1-1.1 [15.2 MB]
Fetched 87.4 MB in 1s (77.1 MB/s)
Selecting previously unselected package conntrack.
(Reading database ... 68011 files and directories currently installed.)
Preparing to unpack .../0-conntrack_1%3a1.4.8-1ubuntu1_amd64.deb ...
Unpacking conntrack (1:1.4.8-1ubuntu1) ...
Selecting previously unselected package cri-tools.
Preparing to unpack .../1-cri-tools_1.31.1-1.1_amd64.deb ...
Unpacking cri-tools (1.31.1-1.1) ...
Selecting previously unselected package kubeadm.
Preparing to unpack .../2-kubeadm_1.31.1-1.1_amd64.deb ...
Unpacking kubeadm (1.31.1-1.1) ...
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
Unpacking cri-tools (1.31.1-1.1) ...
Selecting previously unselected package kubeadm.
Preparing to unpack .../2-kubeadm_1.31.1-1.1_amd64.deb ...
Unpacking kubeadm (1.31.1-1.1) ...
Selecting previously unselected package kubectl.
Preparing to unpack .../3-kubectl_1.31.1-1.1_amd64.deb ...
Unpacking kubectl (1.31.1-1.1) ...
Selecting previously unselected package kubernetes-cni.
Preparing to unpack .../4-kubernetes-cni_1.5.1-1.1_amd64.deb ...
Unpacking kubernetes-cni (1.5.1-1.1) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../5-kubelet_1.31.1-1.1_amd64.deb ...
Unpacking kubelet (1.31.1-1.1) ...
Setting up conntrack (1:1.4.8-1ubuntu1) ...
Setting up kubectl (1.31.1-1.1) ...
Setting up cri-tools (1.31.1-1.1) ...
Setting up kubernetes-cni (1.5.1-1.1) ...
Setting up kubeadm (1.31.1-1.1) ...
Setting up kubelet (1.31.1-1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
```

sudo systemctl enable --now kubelet
sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-20-171:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
W0915 07:47:37.419191    7952 checks.go:1080] [preflight] WARNING: Couldn't create the interface used for talking to the container runtime: failed to create
 new CRI runtime service: validate service connection: validate CRI v1 runtime API for endpoint "unix:///var/run/containerd/containerd.sock": rpc error: cod
e = Unimplemented desc = unknown service runtime.v1.RuntimeService
        [WARNING FileExisting-socat]: socat not found in system path
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
error execution phase preflight: [preflight] Some fatal errors occurred:
failed to create new CRI runtime service: validate service connection: validate CRI v1 runtime API for endpoint "unix:///var/run/containerd/containerd.sock"
: rpc error: code = Unimplemented desc = unknown service runtime.v1.RuntimeService[preflight] If you know what you are doing, you can make a check non-fatal
 with `--ignore-preflight-errors=...`
To see the stack trace of this error execute with --v=5 or higher
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Now We have got an error.
So we have to perform some additional commands as follow.

sudo apt-get install -y containerd

```
To see the stack trace of this error execute with --v=5 or higher   ubuntu@ip-172-31-20-171:~$ sudo apt-get install -y containerd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  runc
The following packages will be REMOVED:
  containerd.io docker-ce
The following NEW packages will be installed:
  containerd runc
0 upgraded, 2 newly installed, 2 to remove and 130 not upgraded.
Need to get 47.2 MB of archives.
After this operation, 53.1 MB disk space will be freed.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Fetched 47.2 MB in 1s (74.5 MB/s)
(Reading database ... 68068 files and directories currently installed.)
Removing docker-ce (5:27.2.1-1~ubuntu.24.04~noble) ...
Removing containerd.io (1.7.22-1) ...
Selecting previously unselected package runc.
(Reading database ... 68048 files and directories currently installed.)
Preparing to unpack .../runc_1.1.12-0ubuntu3.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu3.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../containerd_1.7.12-0ubuntu4.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu4.1) ...
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.
```

```
Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on
 this host.
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml

```
ubuntu@ip-172-31-20-171:~$ sudo mkdir -p /etc/containerd
sudo containerd config default | sudo tee /etc/containerd/config.toml
disabled_plugins = []
imports = []
oom_score = 0
plugin_dir = ""
required_plugins = []
root = "/var/lib/containerd"
state = "/run/containerd"
temp = ""
version = 2

[cgroup]
  path = ""

[debug]
  address = ""
  format = ""
  gid = 0
  level = ""
  uid = 0

[grpc]
  address = "/run/containerd/containerd.sock"
  gid = 0
  max_recv_message_size = 16777216
  max_send_message_size = 16777216
  tcp_address = ""
  tcp_tls_ca = ""
  tcp_tls_cert = ""
  tcp_tls_key = ""
  uid = 0

[metrics]
  address = ""
  grpc_histogram = false

[plugins]

  [plugins."io.containerd.gc.v1.scheduler"]
    deletion_threshold = 0
```

...
sudo systemctl restart containerd
sudo systemctl enable containerd
sudo systemctl status containerd

```
ubuntu@ip-172-31-20-171:~$ sudo systemctl restart containerd
sudo systemctl enable containerd
ubuntu@ip-172-31-20-171:~$ sudo systemctl status containerd
● containerd.service - containerd container runtime
     Loaded: loaded (/usr/lib/systemd/system/containerd.service; en>
     Active: active (running) since Sun 2024-09-15 07:49:23 UTC; 5s>
       Docs: https://containerd.io
   Main PID: 8398 (containerd)
      Tasks: 7
     Memory: 13.5M (peak: 14.0M)
        CPU: 70ms
     CGroup: /system.slice/containerd.service
             └─8398 /usr/bin/containerd

Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
Sep 15 07:49:23 ip-172-31-20-171 systemd[1]: Started containerd.ser>
Sep 15 07:49:23 ip-172-31-20-171 containerd[8398]: time="2024-09-15>
```

sudo apt-get install -y socat

```
ubuntu@ip-172-31-20-171:~$ sudo apt-get install -y socat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libltdl7 libslirp0 pigz slirp4netns
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  socat
0 upgraded, 1 newly installed, 0 to remove and 130 not upgraded.
Need to get 374 kB of archives.
After this operation, 1649 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 socat amd64 1.8.0.0-4build3 [374 kB]
Fetched 374 kB in 0s (12.1 MB/s)
Selecting previously unselected package socat.
(Reading database ... 68112 files and directories currently installed.)
Preparing to unpack .../socat_1.8.0.0-4build3_amd64.deb ...
Unpacking socat (1.8.0.0-4build3) ...
Setting up socat (1.8.0.0-4build3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on
 this host.
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 6: Initialize the Kubecluster

sudo kubeadm init --pod-network-cidr=10.244.0.0/16

```
ubuntu@ip-172-31-20-171:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action beforehand using 'kubeadm config images pull'
W0915 07:49:42.979051    8570 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm.It is recommende
d to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [ip-172-31-20-171 kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.3
1.20.171]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-20-171 localhost] and IPs [172.31.20.171 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-20-171 localhost] and IPs [172.31.20.171 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
[control-plane] Creating static Pod manifest for "kube-scheduler"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Starting the kubelet
[wait-control-plane] Waiting for the kubelet to boot up the control plane as static Pods from directory "/etc/kubernetes/manifests"
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 502.777379ms
[api-check] Waiting for a healthy API server. This can take up to 4m0s
[api-check] The API server is healthy after 4.501245501s
[upload-config] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[kubelet] Creating a ConfigMap "kubelet-config" in namespace kube-system with the configuration for the kubelets in the cluster
[upload-certs] Skipping phase. Please see --upload-certs
[mark-control-plane] Marking the node ip-172-31-20-171 as control-plane by adding the labels: [node-role.kubernetes.io/control-plane node.kubernetes.io/exclude-from-external-load-balancers
]
[mark-control-plane] Marking the node ip-172-31-20-171 as control-plane by adding the taints [node-role.kubernetes.io/control-plane:NoSchedule]
[bootstrap-token] Using token: 7acddu.inheshzwxti0372v
```

```
[mark-control-plane] Marking the node ip-172-31-20-171 as control-plane by adding the taints [node-role.kubernetes.io/control-plane:NoSchedule]
[bootstrap-token] Using token: 7acddu.inheshzwxti0372v
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.20.171:6443 --token 7acddu.inheshzwxti0372v \
        --discovery-token-ca-cert-hash sha256:aed5faf97bac361d1bb7f33a89fb05d2bb28c7fc065024eac2302a734c330a36
```

Copy the mkdir and chown commands from the top and execute them.

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

```
ubuntu@ip-172-31-20-171:~$  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Add a common networking plugin called flannel as mentioned in the code.
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml

```
ubuntu@ip-172-31-20-171:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

Step 7: Now that the cluster is up and running, we can deploy our nginx server on this cluster.Apply this deployment file using this command to create a deployment

kubectl apply -f https://k8s.io/examples/application/deployment.yaml

```
ubuntu@ip-172-31-20-171:~$ kubectl apply -f https://k8s.io/examples/application/deployment.yaml
deployment.apps/nginx-deployment created
```

kubectl get pods

```
ubuntu@ip-172-31-20-171:~$ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-vz8rv   0/1     Pending   0          8s
nginx-deployment-d556bf558-wz5wc   0/1     Pending   0          8s
```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")

kubectl port-forward $POD_NAME 8080:80

```
ubuntu@ip-172-31-20-171:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
ubuntu@ip-172-31-20-171:~$ kubectl port-forward $POD_NAME 8080:80
error: unable to forward port because pod is not running. Current status=Pending
```

Note : We have faced an error as pod status is pending so make it running run below commands then again run above 2 commands.

kubectl taint nodes --all node-role.kubernetes.io/control-plane-node/ip-172-31-20-171 untainted

kubectl get nodes

```
ubuntu@ip-172-31-20-171:~$ kubectl taint nodes --all node-role.kubernetes.io/control-plane-
node/ip-172-31-20-171 untainted
ubuntu@ip-172-31-20-171:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE     VERSION
ip-172-31-20-171   Ready    control-plane   5m23s   v1.31.1
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

kubectl get pods

```
ubuntu@ip-172-31-20-171:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-d556bf558-vz8rv    1/1     Running   0          3m4s
nginx-deployment-d556bf558-wz5wc    1/1     Running   0          3m4s
```

POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")

kubectl port-forward $POD_NAME 8080:80

```
ubuntu@ip-172-31-20-171:~$ POD_NAME=$(kubectl get pods -l app=nginx -o jsonpath="{.items[0].metadata.name}")
kubectl port-forward $POD_NAME 8080:80
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
Handling connection for 8080
```

Step 8: Verify your deployment
Open up a new terminal and ssh to your EC2 instance.
Then, use this curl command to check if the Nginx server is running.

curl --head http://127.0.0.1:8080

```
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\bhush\one drive 2\OneDrive\Desktop\New folder (4)> ssh -i "Master_Ec2_Key.pem" ubuntu@ec2-54-196-129-215.compute-1.amazonaws.com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

 System information as of Sun Sep 15 07:58:53 UTC 2024

  System load:  0.15              Processes:             152
  Usage of /:   55.3% of 6.71GB   Users logged in:       1
  Memory usage: 20%               IPv4 address for enX0: 172.31.20.171
  Swap usage:   0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

132 updates can be applied immediately.
38 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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


Last login: Sun Sep 15 07:27:23 2024 from 152.58.2.47
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
ubuntu@ip-172-31-20-171:~$ curl --head http://127.0.0.1:8080
HTTP/1.1 200 OK
Server: nginx/1.14.2
Date: Sun, 15 Sep 2024 07:59:03 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Tue, 04 Dec 2018 14:44:49 GMT
Connection: keep-alive
ETag: "5c0692e1-264"
Accept-Ranges: bytes

ubuntu@ip-172-31-20-171:~$
```

If the response is 200 OK and you can see the Nginx server name, your deployment was successful.
We have successfully deployed our Nginx server on our EC2 instance.

Conclusion:
In this experiment, we successfully installed Kubernetes on an EC2 instance and deployed an Nginx server using Kubectl commands. During the process, we encountered two main errors: the Kubernetes pod was initially in a pending state, which was resolved by removing the control-plane taint using kubectl taint nodes --all, and we also faced an issue with the missing containerd runtime, which was fixed by installing and starting containerd. We used a **t2.medium EC2 instance with 2 CPUs** to meet the necessary resource requirements for the Kubernetes setup and deployment.

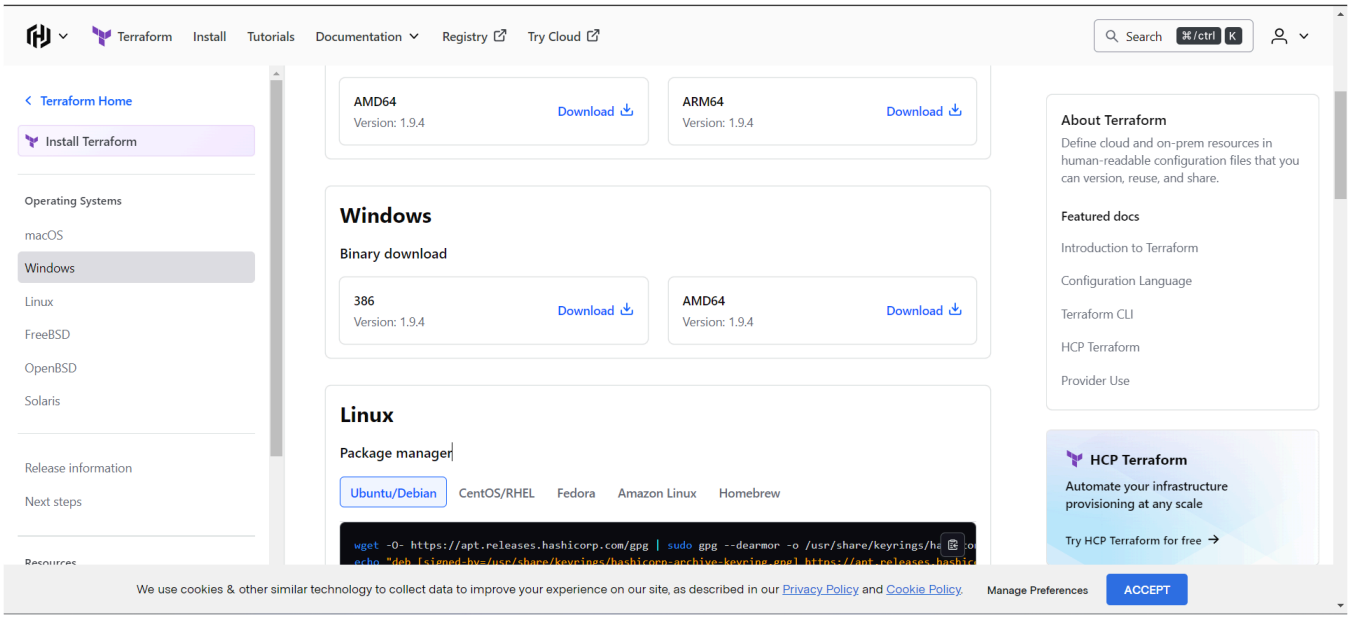
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To understand terraform lifecycle, and core concepts/terminologies, and install it on a Linux Machine or Windows.

Terraform: Terraform is an open-source Infrastructure as Code (IaC) tool developed by HashiCorp. It allows you to define, provision, and manage infrastructure across various cloud providers using a simple declarative configuration language. Terraform's key features include automation of infrastructure deployments, version control for infrastructure changes, and the ability to manage both cloud and on-premises resources in a consistent manner.

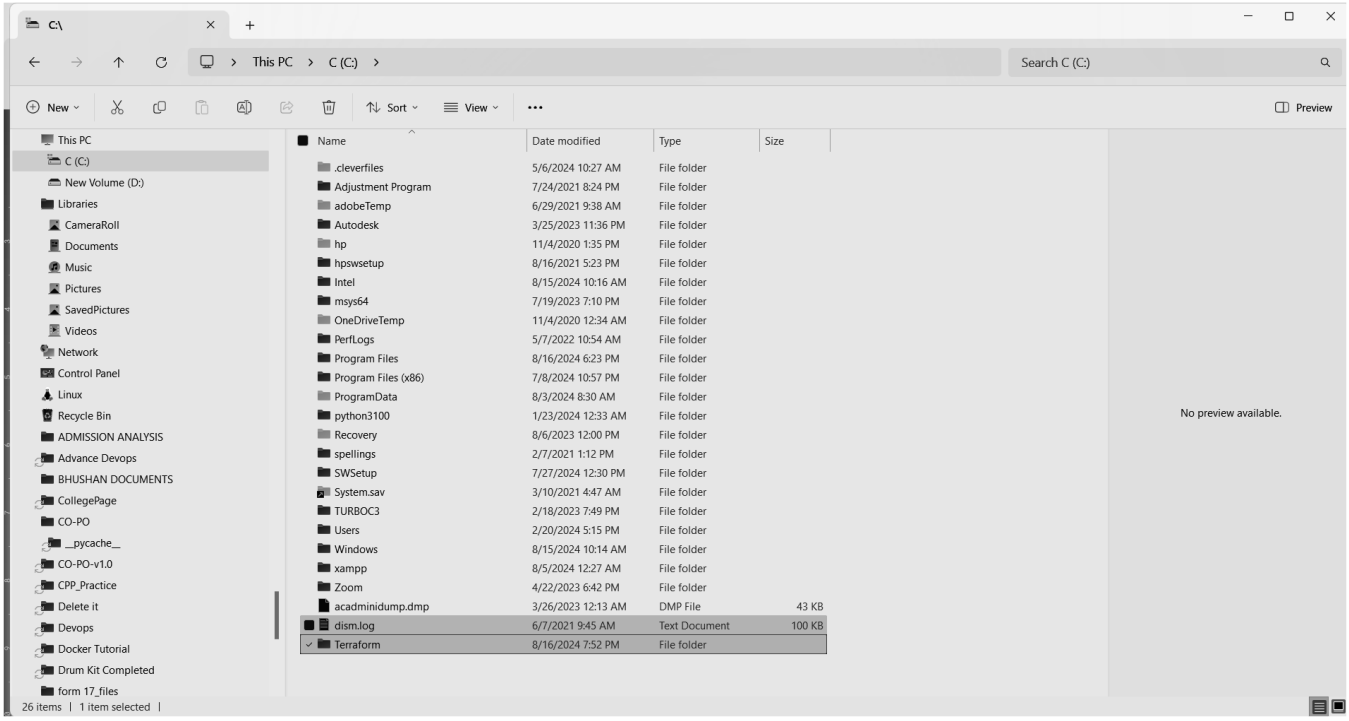
Step 1: Visit the website: https://developer.hashicorp.com/terraform/install?product_intent=terraform and install the terraform according to your system. Here we are downloading it for Windows.

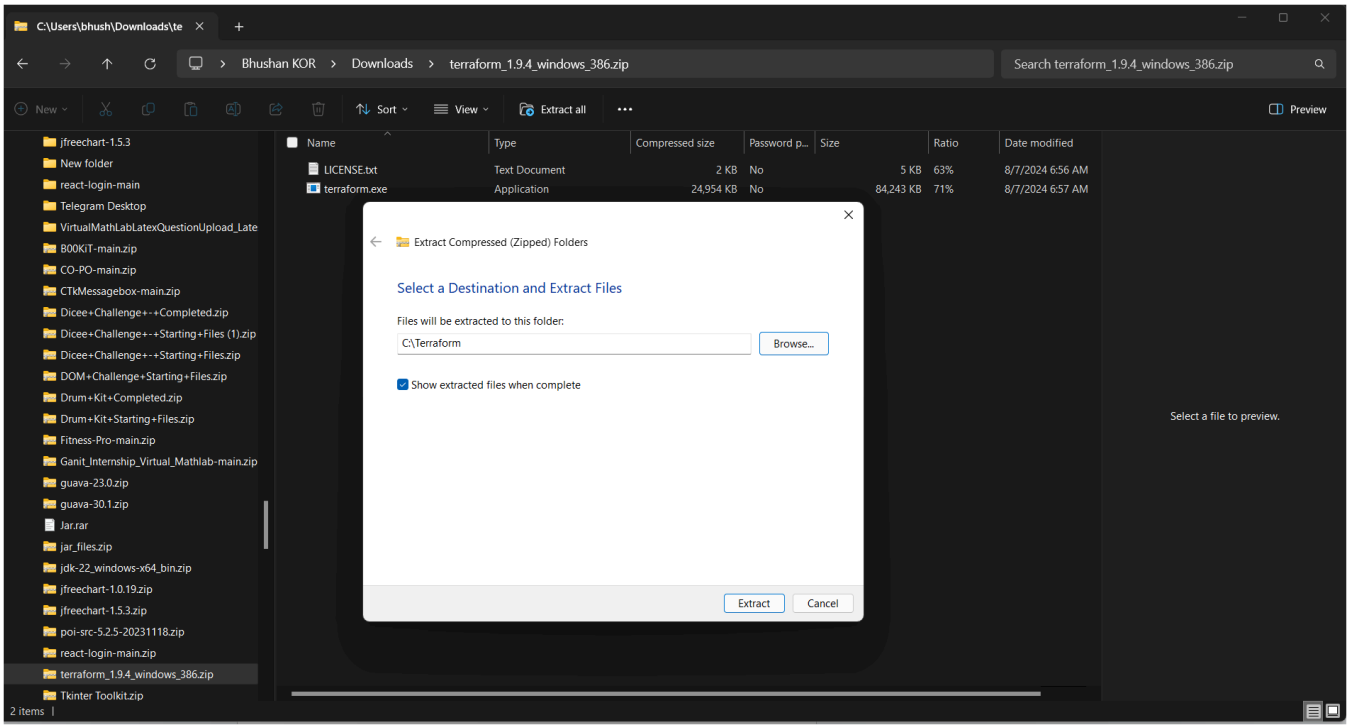


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 2: Done Downloaded Successfully Now create the folder named Terraform in C: directory and extract the file in C:\Terraform.

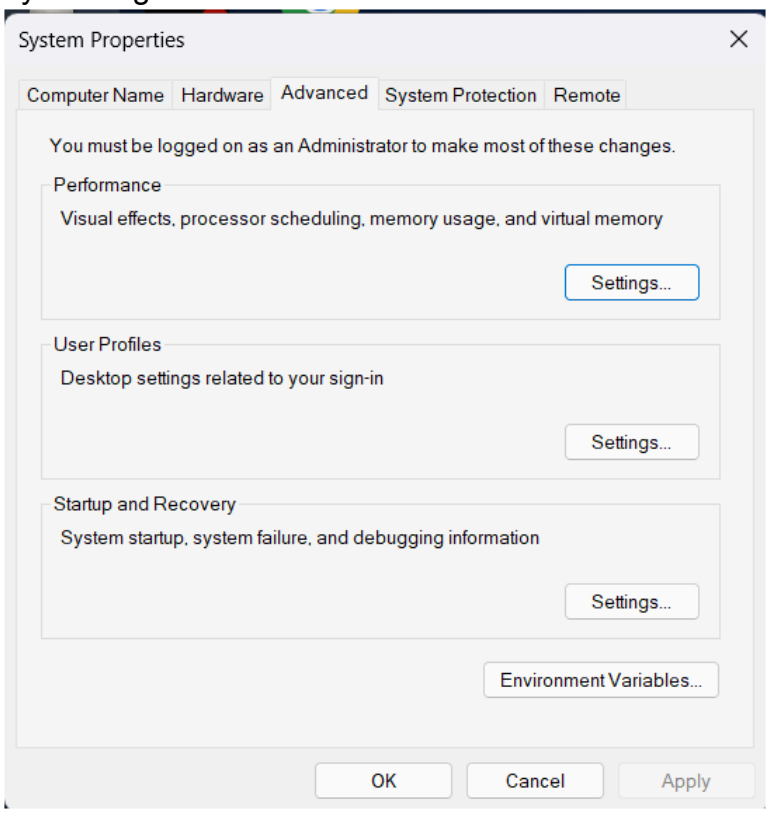


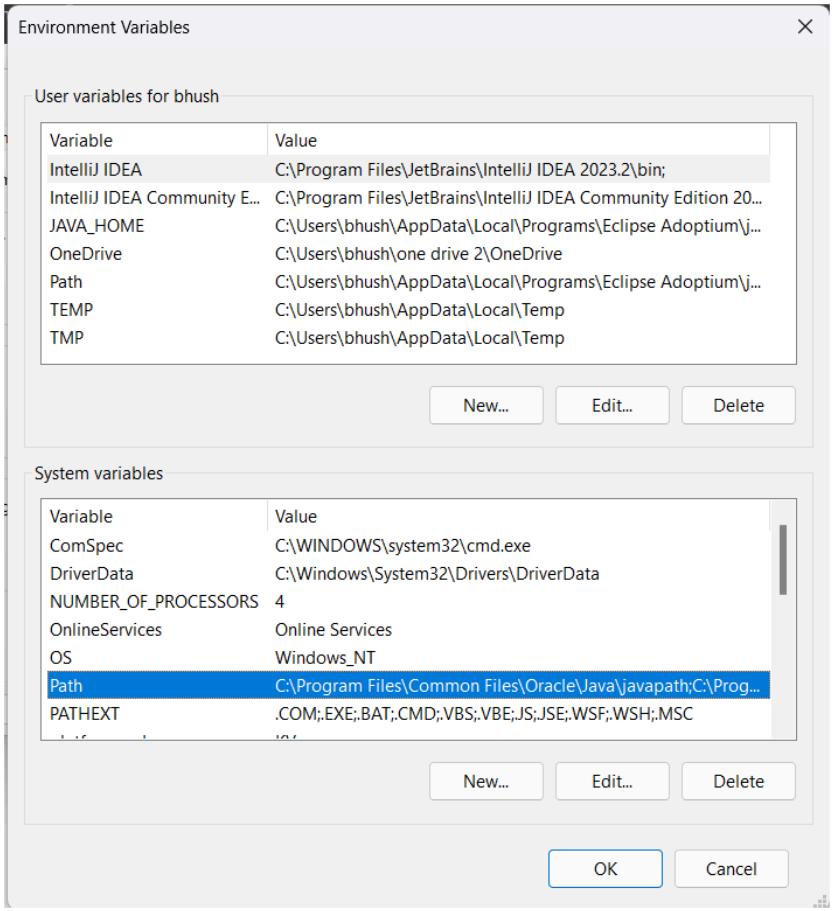


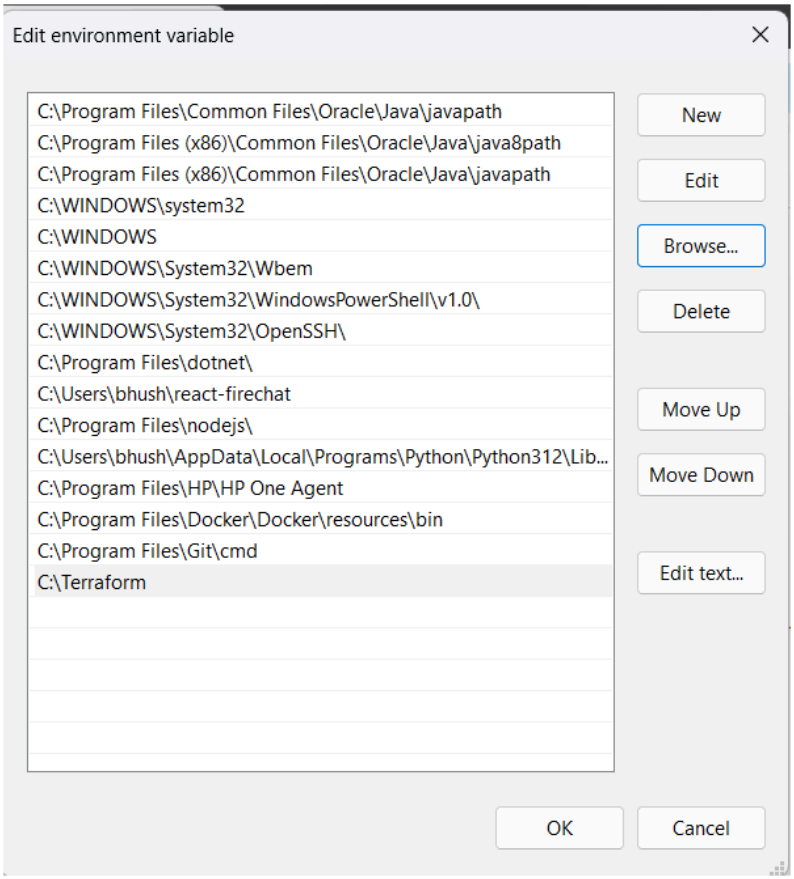
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

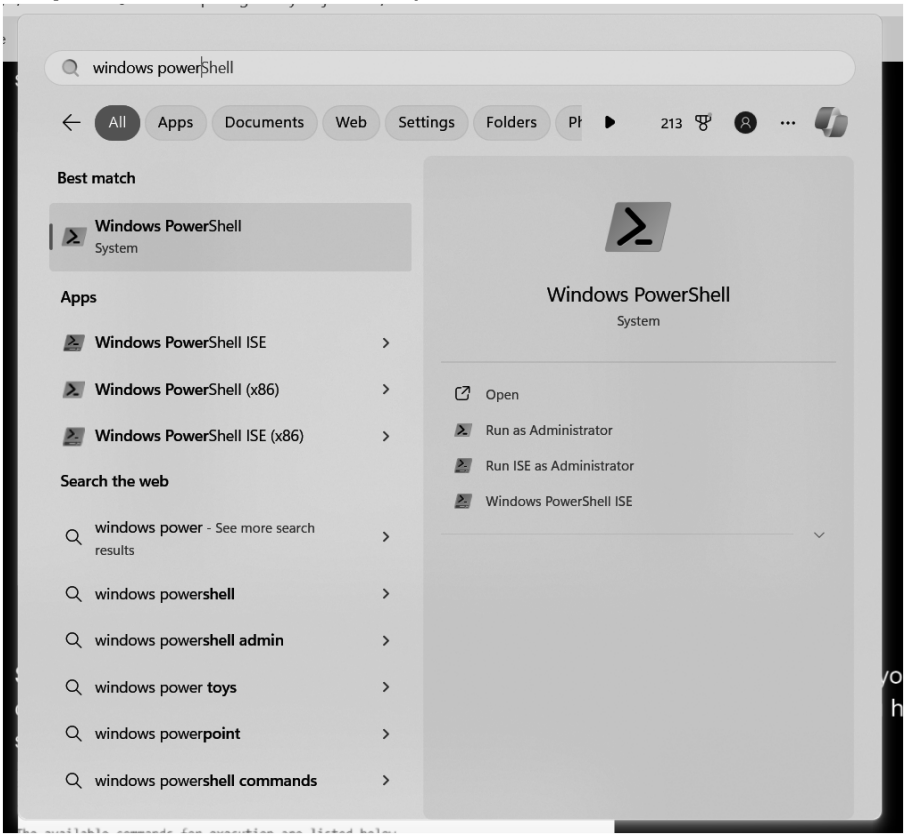
Step 3: Edit the environment variable. In the path of system variables add the path of terraform by clicking edit. After that save and close it.



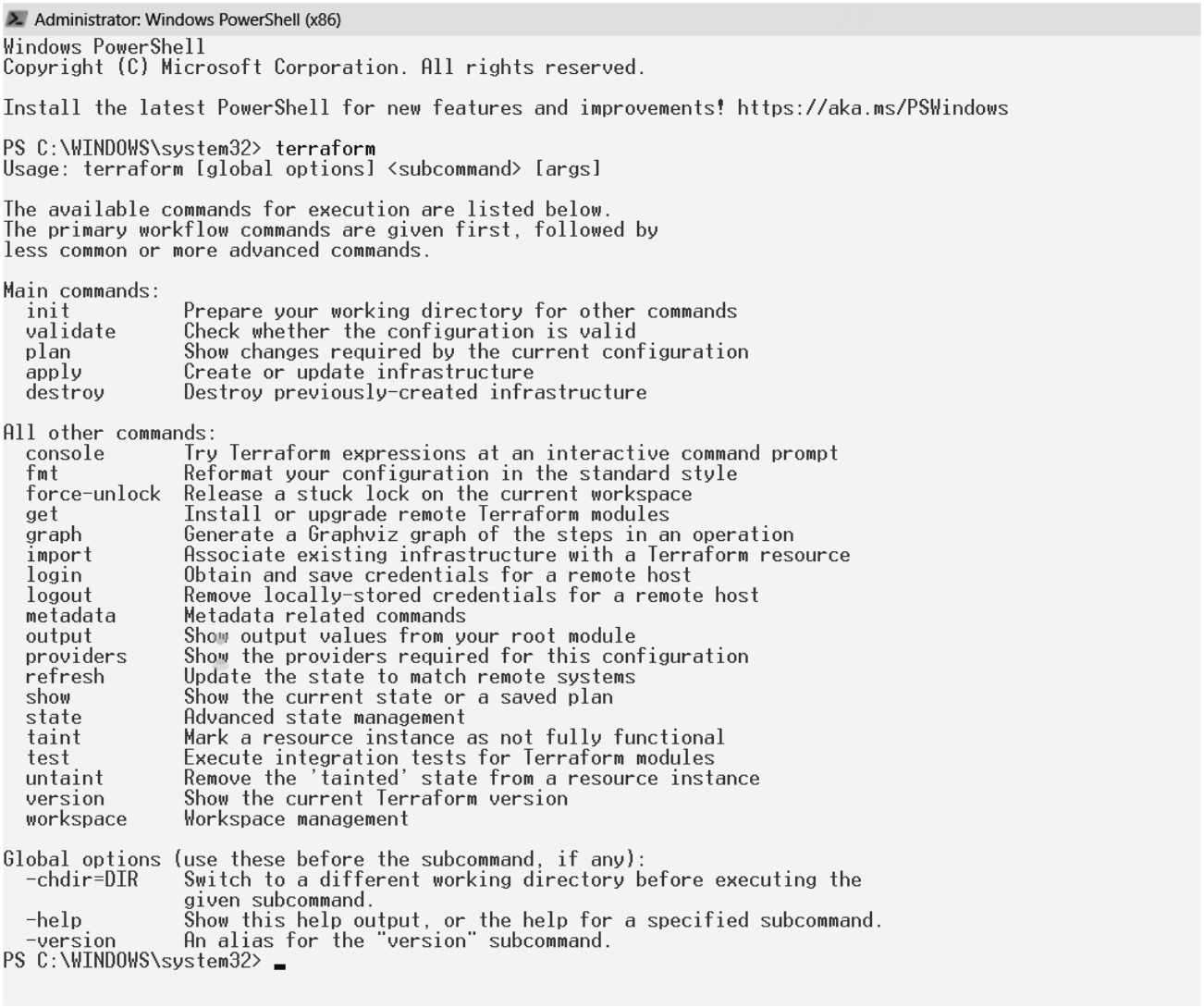




Step 4: Search for Windows power shell and Run it as administrator.



Step 5: Now run the command terraform in Windoes Powershell.



Step 6: Congratulations You have successfully installed Terraform in your system.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform. (S3 bucket or Docker)fdp.

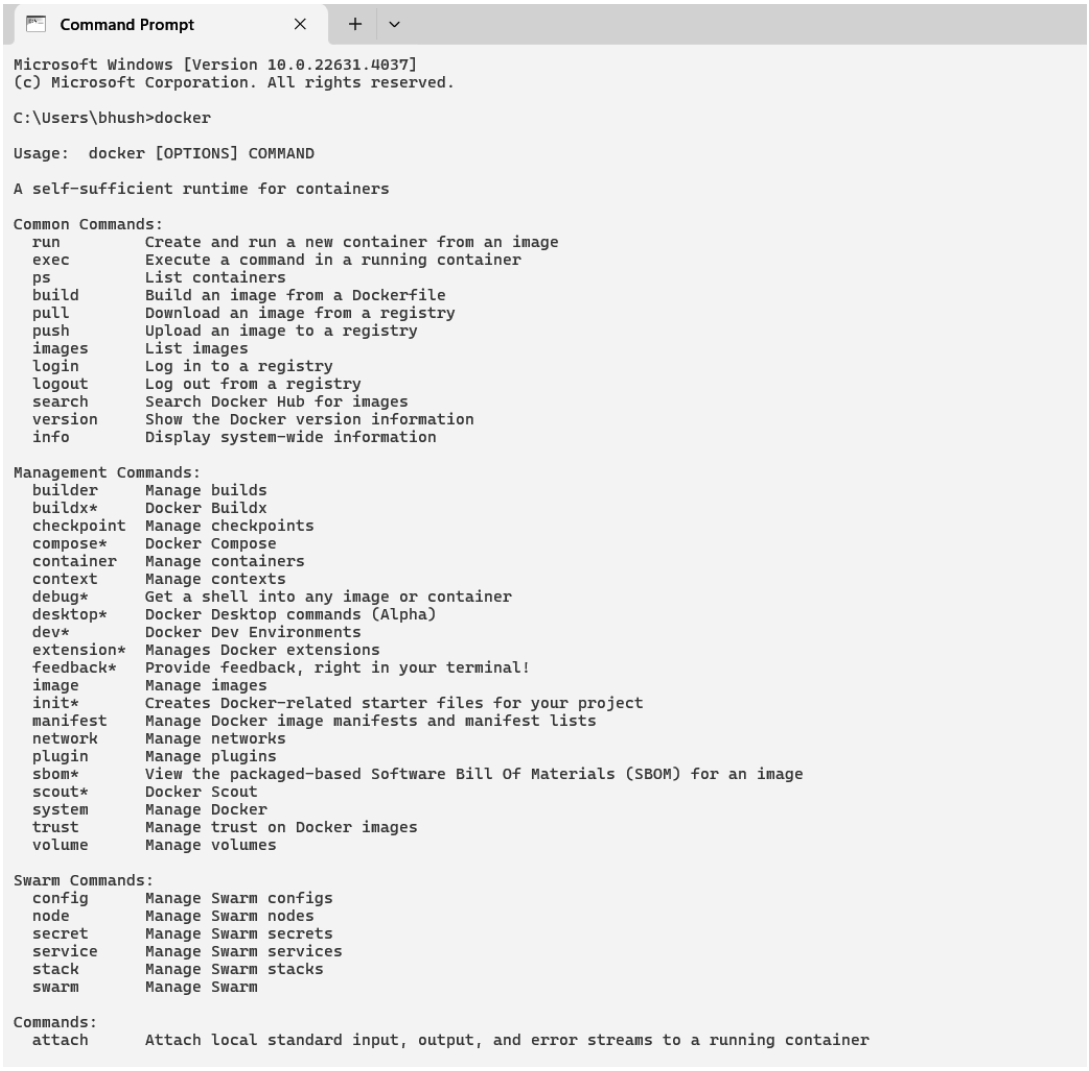
Terraform: Terraform is an open-source infrastructure as code (IaC) tool that allows you to define, provision, and manage cloud resources across various providers using a declarative configuration language. It enables consistent and repeatable infrastructure deployments, supports multi-cloud environments, and maintains state files to track resource changes. Terraform automates the creation and management of infrastructure, making it easier to scale and modify resources.

Creating a docker image using Terraform :

Prerequisite:

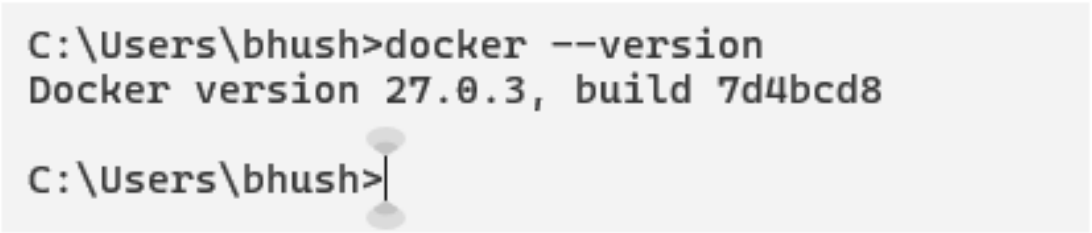
Download and Install Docker Desktop from https://www.docker.com/

Step 1: Run docker command in cmd to check the functionality of docker and also run docker --version to check which docker version is installed on your system.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



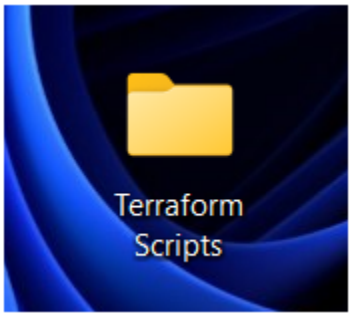
Now, create a folder named ‘Terraform Scripts’ in which we save our different types of scripts which will be further used in this experiment.

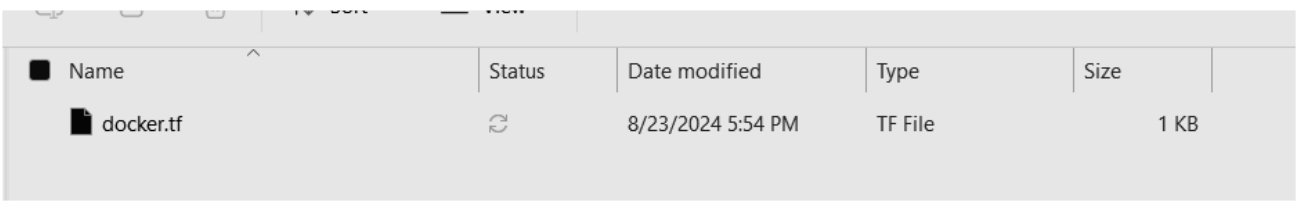
Step 2: Now create a new folder named ‘Docker’ in the ‘TerraformScripts’ folder. Then create a new docker.tf file using Atom editor or Vscode and write the following contents into it to create a Ubuntu Linux container.

(Note part of the script highlighted is a must to get the image otherwise it will give an error.)

Script:

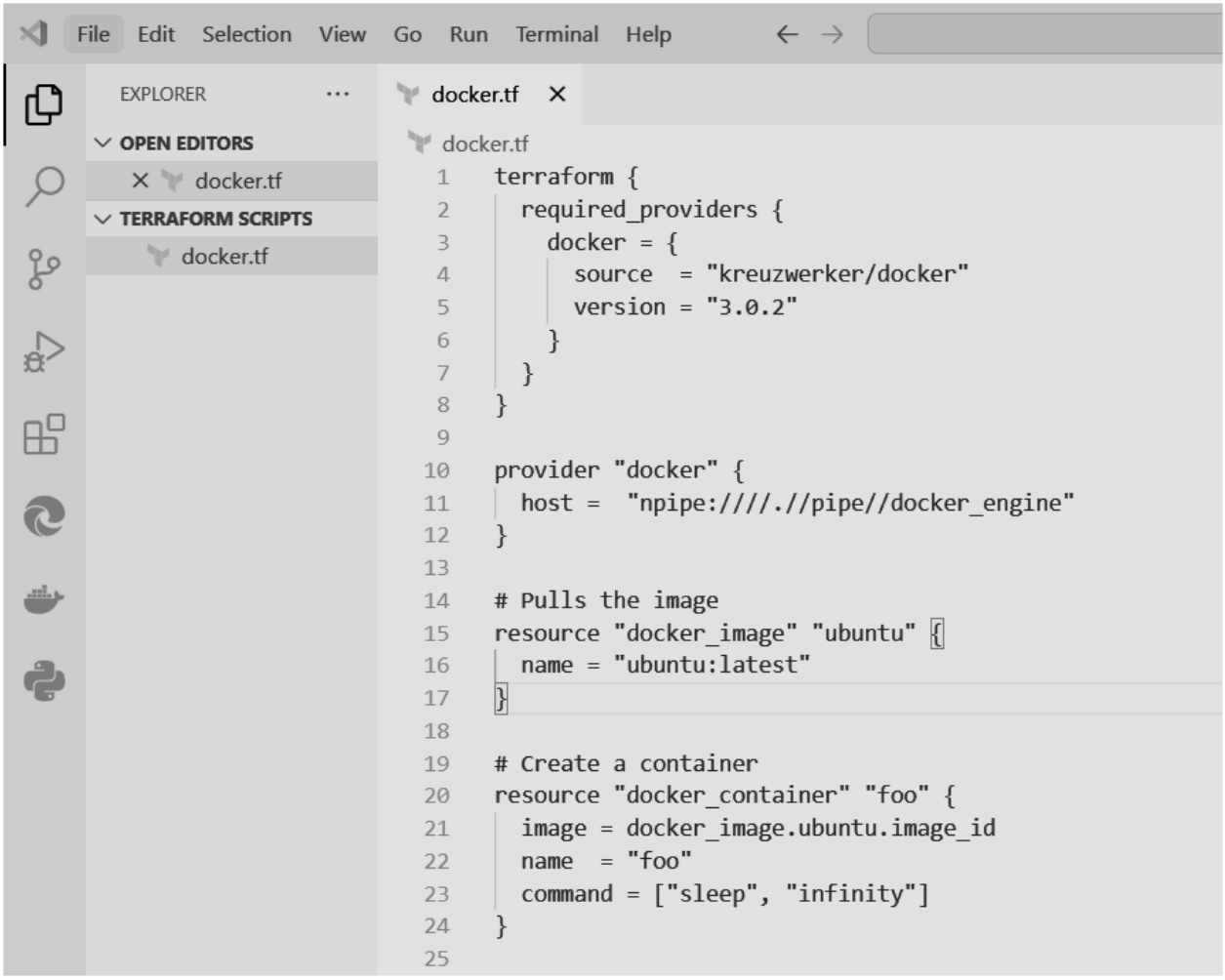
```
terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "3.0.2"
    }
  }
}

provider "docker" {
  host =  "npipe:////.//pipe//docker_engine"
}

# Pulls the image
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name  = "foo"
  command = ["sleep", "infinity"]
}
```



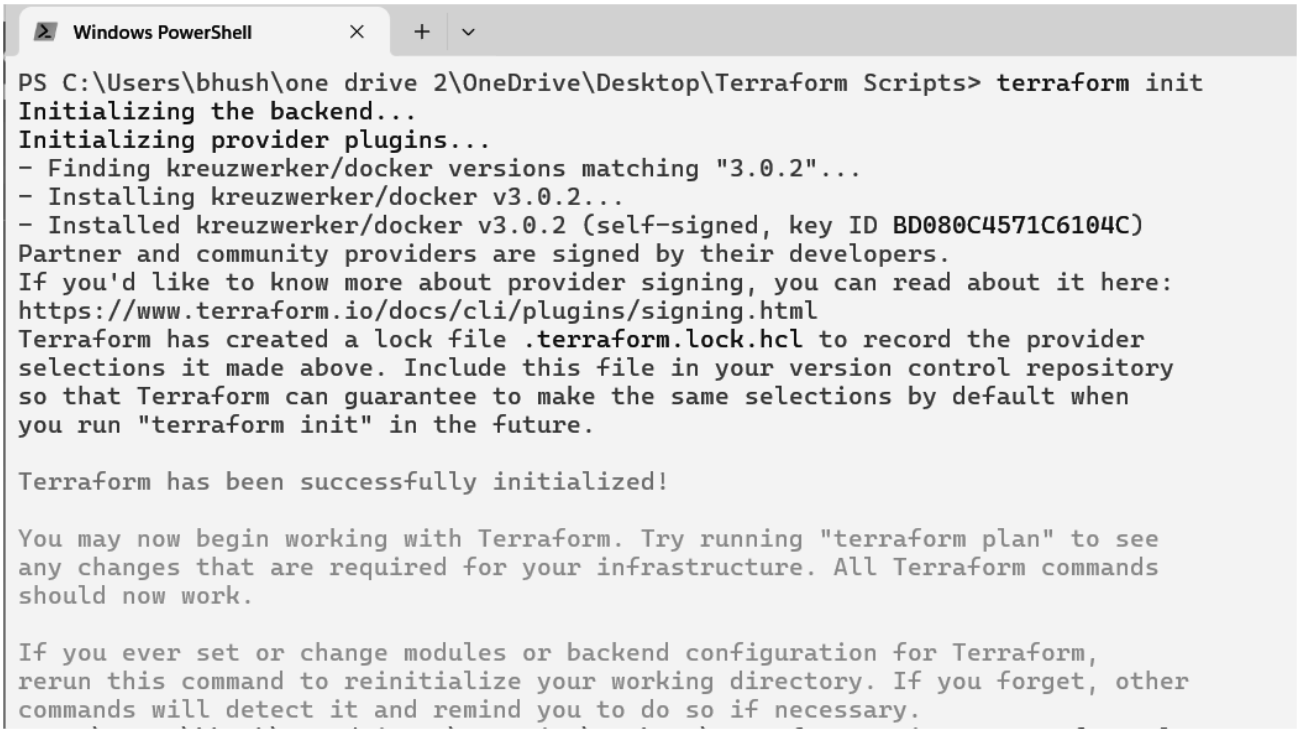


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



Step 3: Now open the terminal in the Terraform Scripts folder and Execute the terraform init command to initialize resources. This will initialize terraform in directory.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 4: Run the command terraform plan. This will create an execution plan and let you overview changes that are going to happen in your infrastructure.

```
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts> terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach                                      = false
      + bridge                                      = (known after apply)
      + command                                     = [
          + "sleep",
          + "infinity",
        ]
      + container_logs                              = (known after apply)
      + container_read_refresh_timeout_milliseconds = 15000
      + entrypoint                                  = (known after apply)
      + env                                         = (known after apply)
      + exit_code                                   = (known after apply)
      + hostname                                    = (known after apply)
      + id                                          = (known after apply)
      + image                                       = (known after apply)
      + init                                        = (known after apply)
      + ipc_mode                                    = (known after apply)
      + log_driver                                  = (known after apply)
      + logs                                        = false
      + must_run                                    = true
      + name                                        = "foo"
      + network_data                                = (known after apply)
      + read_only                                   = false
      + remove_volumes                              = true
      + restart                                     = "no"
      + rm                                          = false
      + runtime                                     = (known after apply)
      + security_opts                               = (known after apply)
      + shm_size                                    = (known after apply)
      + start                                       = true
      + stdin_open                                  = false
      + stop_signal                                 = (known after apply)
      + stop_timeout                                = (known after apply)
      + tty                                         = false
      + wait                                        = false
      + wait_timeout                                = 60

      + healthcheck (known after apply)

      + labels (known after apply)
    }

  # docker_image.ubuntu will be created
  + resource "docker_image" "ubuntu" {
      + id          = (known after apply)
      + image_id    = (known after apply)
      + name        = "ubuntu:latest"
      + repo_digest = (known after apply)
    }

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

---

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if
you run "terraform apply" now.
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 5: Now, run the command terraform apply to carry out the changes that
We have made when terrafrom plan command was executed.After running the command it will ask for a value for confirmation that time type yes.(Before step 5 run command docker images for next step)

```
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # docker_container.foo will be created
  + resource "docker_container" "foo" {
      + attach                                      = false
      + bridge                                      = (known after apply)
      + command                                     = [
          + "sleep",
          + "infinity",
        ]
      + container_logs                              = (known after apply)
      + container_read_refresh_timeout_milliseconds = 15000
      + entrypoint                                  = (known after apply)
      + env                                         = (known after apply)
      + exit_code                                   = (known after apply)
      + hostname                                    = (known after apply)
      + id                                          = (known after apply)
      + image                                       = (known after apply)
      + init                                        = (known after apply)
      + ipc_mode                                    = (known after apply)
      + log_driver                                  = (known after apply)
      + logs                                        = false
      + must_run                                    = true
      + name                                        = "foo"
      + network_data                                = (known after apply)
      + read_only                                   = false
      + remove_volumes                              = true
      + restart                                     = "no"
      + rm                                          = false
      + runtime                                     = (known after apply)
      + security_opts                               = (known after apply)
      + shm_size                                    = (known after apply)
      + start                                       = true
      + stdin_open                                  = false
      + stop_signal                                 = (known after apply)
      + stop_timeout                                = (known after apply)
      + tty                                         = false
      + wait                                        = false
      + wait_timeout                                = 60

      + healthcheck (known after apply)

      + labels (known after apply)
    }

  # docker_image.ubuntu will be created
  + resource "docker_image" "ubuntu" {
      + id          = (known after apply)
      + image_id    = (known after apply)
      + name        = "ubuntu:latest"
      + repo_digest = (known after apply)
    }

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

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

  Enter a value: yes

docker_image.ubuntu: Creating...
docker_image.ubuntu: Still creating... [10s elapsed]
docker_image.ubuntu: Still creating... [20s elapsed]
docker_image.ubuntu: Still creating... [30s elapsed]
docker_image.ubuntu: Still creating... [40s elapsed]
docker_image.ubuntu: Creation complete after 42s [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Creating...
docker_container.foo: Creation complete after 3s [id=2157b0fa12aed015eaaf4b3686ce28eca721f409d1256450da2512f0830374c3]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 6: Now run the command docker images before terraform apply command and after terraform apply command.And see the changes.

Before Terraform apply Command :

```
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts>
```

After Terraform apply Command :

```
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts> docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    edbfe74c41f8   3 weeks ago   78.1MB
```

Step 7: From above command we can clearly see that the ubuntu image is created.Now we have to destroy it, so we will use terraform destroy command.After running the command it will ask for a value for confirmation that time type yes.

```
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts> terraform destroy
docker_image.ubuntu: Refreshing state... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_container.foo: Refreshing state... [id=2157b0fa12aed015eaaf4b3686ce28eca721f409d1256450da2512f0830374c3]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  - destroy

Terraform will perform the following actions:

  # docker_container.foo will be destroyed
  - resource "docker_container" "foo" {
      - attach                                      = false -> null
      - command                                     = [
          - "sleep",
          - "infinity",
        ] -> null
      - container_read_refresh_timeout_milliseconds = 15000 -> null
      - cpu_shares                                  = 0 -> null
      - dns                                         = [] -> null
      - dns_opts                                    = [] -> null
      - dns_search                                  = [] -> null
      - entrypoint                                  = [] -> null
      - env                                         = [] -> null
      - group_add                                   = [] -> null
      - hostname                                    = "2157b0fa12ae" -> null
      - id                                          = "2157b0fa12aed015eaaf4b3686ce28eca721f409d1256450da2512f0830374c3"
 -> null
      - image                                       = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - init                                        = false -> null
      - ipc_mode                                    = "private" -> null
      - log_driver                                  = "json-file" -> null
      - log_opts                                    = {} -> null
      - logs                                        = false -> null
      - max_retry_count                             = 0 -> null
      - memory                                      = 0 -> null
      - memory_swap                                 = 0 -> null
      - must_run                                    = true -> null
      - name                                        = "foo" -> null
      - network_data                                = [
          - {
              - gateway                   = "172.17.0.1"
              - global_ipv6_prefix_length = 0
              - ip_address                = "172.17.0.2"
              - ip_prefix_length          = 16
              - mac_address               = "02:42:ac:11:00:02"
              - network_name              = "bridge"
                # (2 unchanged attributes hidden)
            },
        ] -> null
      - network_mode                                = "bridge" -> null
      - privileged                                  = false -> null
      - publish_all_ports                           = false -> null
      - read_only                                   = false -> null
      - remove_volumes                              = true -> null
      - restart                                     = "no" -> null
```

```
      - remove_volumes                              = true -> null
      - restart                                     = "no" -> null
      - rm                                          = false -> null
      - runtime                                     = "runc" -> null
      - security_opts                               = [] -> null
      - shm_size                                    = 64 -> null
      - start                                       = true -> null
      - stdin_open                                  = false -> null
      - stop_timeout                                = 0 -> null
      - storage_opts                                = {} -> null
      - sysctls                                     = {} -> null
      - tmpfs                                       = {} -> null
      - tty                                         = false -> null
      - wait                                        = false -> null
      - wait_timeout                                = 60 -> null
        # (8 unchanged attributes hidden)
    }

  # docker_image.ubuntu will be destroyed
  - resource "docker_image" "ubuntu" {
      - id          = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest" -> null
      - image_id    = "sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598a" -> null
      - name        = "ubuntu:latest" -> null
      - repo_digest = "ubuntu@sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee" -> null
    }

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

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

docker_container.foo: Destroying... [id=2157b0fa12aed015eaaf4b3686ce28eca721f409d1256450da2512f0830374c3]
docker_container.foo: Destruction complete after 1s
docker_image.ubuntu: Destroying... [id=sha256:edbfe74c41f8a3501ce542e137cf28ea04dd03e6df8c9d66519b6ad761c2598aubuntu:latest]
docker_image.ubuntu: Destruction complete after 1s

Destroy complete! Resources: 2 destroyed.
```

Again Run docker image command to verify image is deleted or not.

```
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts> docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
PS C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts>
```

Step 7: Done You have successfully created a docker image of Ubuntu using Terraform and also destroyed it.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To Build, change, and destroy AWS / GCP /Microsoft Azure/ DigitalOcean infrastructure Using Terraform. (S3 bucket or Docker)fdp.

Terraform: Terraform is an open-source infrastructure as code (IaC) tool that allows you to define, provision, and manage cloud resources across various providers using a declarative configuration language. It enables consistent and repeatable infrastructure deployments, supports multi-cloud environments, and maintains state files to track resource changes. Terraform automates the creation and management of infrastructure, making it easier to scale and modify resources.

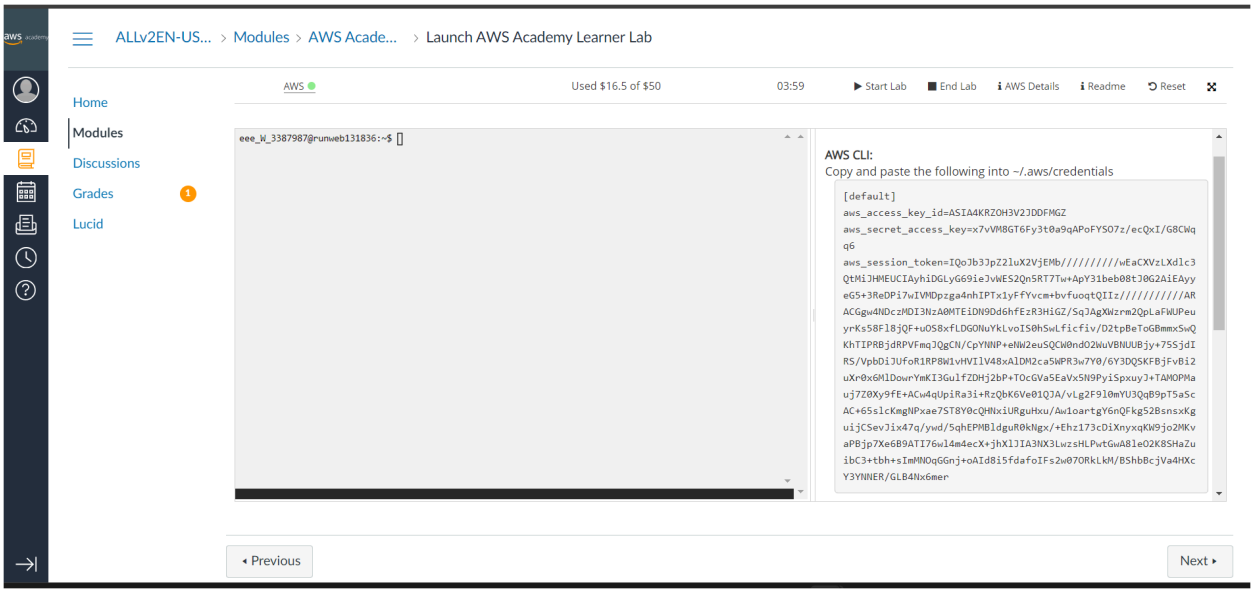
Prerequisites:

- Install Terraform .
- Install Vscode.
- Hashicrop extension in Vscode.
  AWS Academy Account

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 1: Open your AWS Academy account . Then Start the lab from modules. After starting the lab click on the AWS details and click on show button after the AWS CLI to get Access keys and other details now copy full credientials.



Step 2: Now Create a folder Named As “Terraform Scripts “ and Open it on Vscode or any code editor. Now create one file Credietial.txt and paste the copied credentials 2 times .

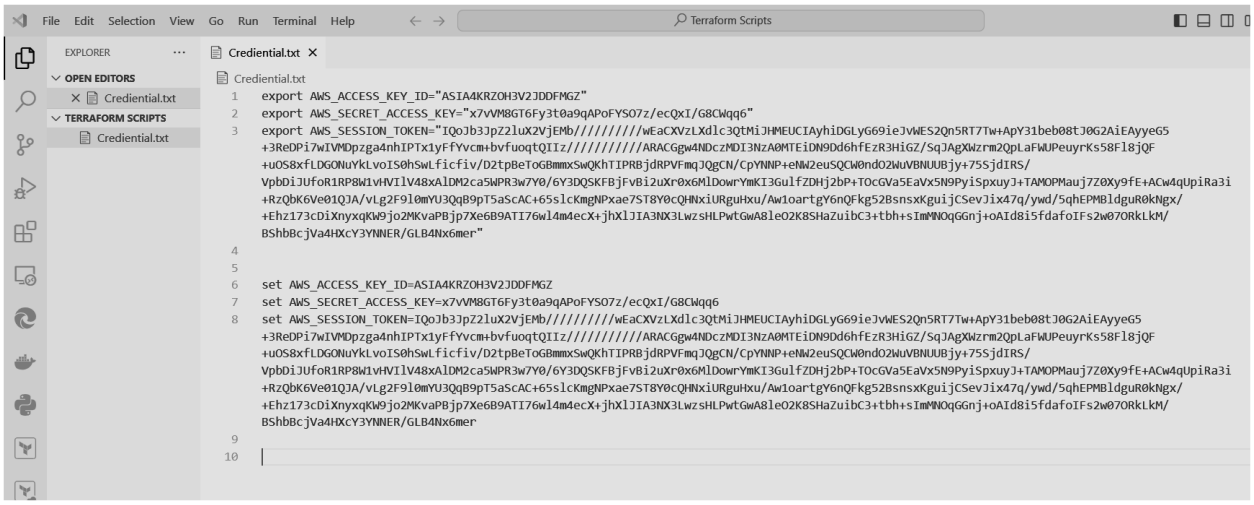
```
aws_access_key_id=ASIA4KRZOH3V2JDDFMGZ
aws_secret_access_key=x7vVM8GT6Fy3t0a9qAPoFYSO7z/ecQxI/G8CWqq6
aws_session_token=IQoJb3JpZ2luX2VjEMb/////////wEaCXVzLXdlc3QtMiJHMEUCIAyhiDGLyG69ieJvWES2Qn5RT7Tw+ApY31beb08tJ0G2AiEAyyeG5
+3ReDPi7wIVMDpzga4nhIPTx1yFfYvcm+bvfuoqtQIIz//////////ARACGgw4NDczMDI3NzA0MTEiDN9Dd6hfEzR3HiGZ/SqJAgXWzrm2QpLaFWUPeuyrKs58Fl8jQF
+uOS8xfLDGONuYkLvoIS0hSwLficfiv/D2tpBeToGBmmxSwQKhTIPRBjdRPVFmqJQgCN/CpYNNP+eNW2euSQCW0ndO2WuVBNUUBjy+75SjdIRS/
VpbDiJUfoR1RP8W1vHVIlV48xAlDM2ca5WPR3w7Y0/6Y3DQSKFBjFvBi2uXr0x6MlDowrYmKI3GulfZDHj2bP+TOcGVa5EaVx5N9PyiSpxuyJ+TAMOPMauj7Z0Xy9fE+ACw4qUpiRa3i
+RzQbK6Ve01QJA/vLg2F9l0mYU3QqB9pT5aScAC+65slcKmgNPxae7ST8Y0cQHNxiURguHxu/Aw1oartgY6nQFkg52BsnsxKguijCSevJix47q/ywd/5qhEPMBldguR0kNgx/
+Ehz173cDiXnyxqKW9jo2MKvaPBjp7Xe6B9ATI76wl4m4ecX+jhXlJIA3NX3LwzsHLPwtGwA8leO2K8SHaZuibC3+tbh+sImMNOqGGnj+oAId8i5fdafoIFs2w07ORkLkM/
BShbBcjVa4HXcY3YNNER/GLB4Nx6mer
```


```
aws_access_key_id=ASIA4KRZOH3V2JDDFMGZ
aws_secret_access_key=x7vVM8GT6Fy3t0a9qAPoFYSO7z/ecQxI/G8CWqq6
aws_session_token=IQoJb3JpZ2luX2VjEMb/////////wEaCXVzLXdlc3QtMiJHMEUCIAyhiDGLyG69ieJvWES2Qn5RT7Tw+ApY31beb08tJ0G2AiEAyyeG5
+3ReDPi7wIVMDpzga4nhIPTx1yFfYvcm+bvfuoqtQIIz//////////ARACGgw4NDczMDI3NzA0MTEiDN9Dd6hfEzR3HiGZ/SqJAgXWzrm2QpLaFWUPeuyrKs58Fl8jQF
+uOS8xfLDGONuYkLvoIS0hSwLficfiv/D2tpBeToGBmmxSwQKhTIPRBjdRPVFmqJQgCN/CpYNNP+eNW2euSQCW0ndO2WuVBNUUBjy+75SjdIRS/
VpbDiJUfoR1RP8W1vHVIlV48xAlDM2ca5WPR3w7Y0/6Y3DQSKFBjFvBi2uXr0x6MlDowrYmKI3GulfZDHj2bP+TOcGVa5EaVx5N9PyiSpxuyJ+TAMOPMauj7Z0Xy9fE+ACw4qUpiRa3i
+RzQbK6Ve01QJA/vLg2F9l0mYU3QqB9pT5aScAC+65slcKmgNPxae7ST8Y0cQHNxiURguHxu/Aw1oartgY6nQFkg52BsnsxKguijCSevJix47q/ywd/5qhEPMBldguR0kNgx/
+Ehz173cDiXnyxqKW9jo2MKvaPBjp7Xe6B9ATI76wl4m4ecX+jhXlJIA3NX3LwzsHLPwtGwA8leO2K8SHaZuibC3+tbh+sImMNOqGGnj+oAId8i5fdafoIFs2w07ORkLkM/
BShbBcjVa4HXcY3YNNER/GLB4Nx6mer
```

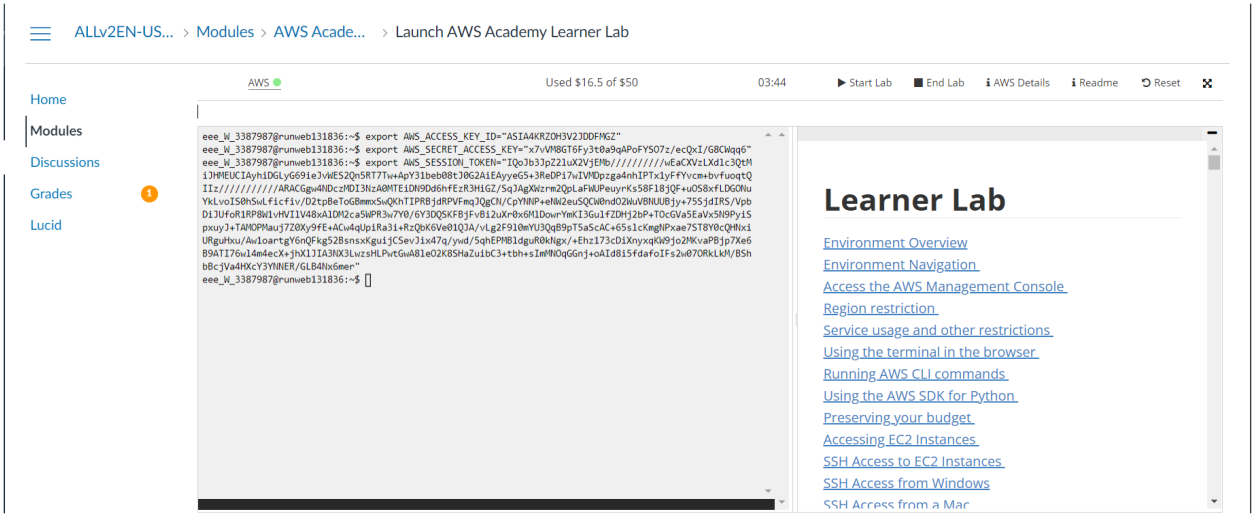

Now Make RHS of both copied keys to upper case and put the LHS of 1st copied keys inside the Double quotes “”
And for 1st copied give prefix export and 2nd copied give prefix set.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

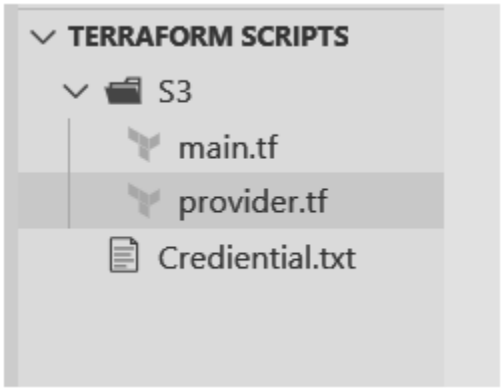
Division: D15C **Roll No: 28**



Step 3: Copy paste the 1st paragraph or 1st copied in learners lab terminal and hit enter.



Step 4: Now create S3 folder inside the same folder and create provider.tf and main.tf file in S3 folder.



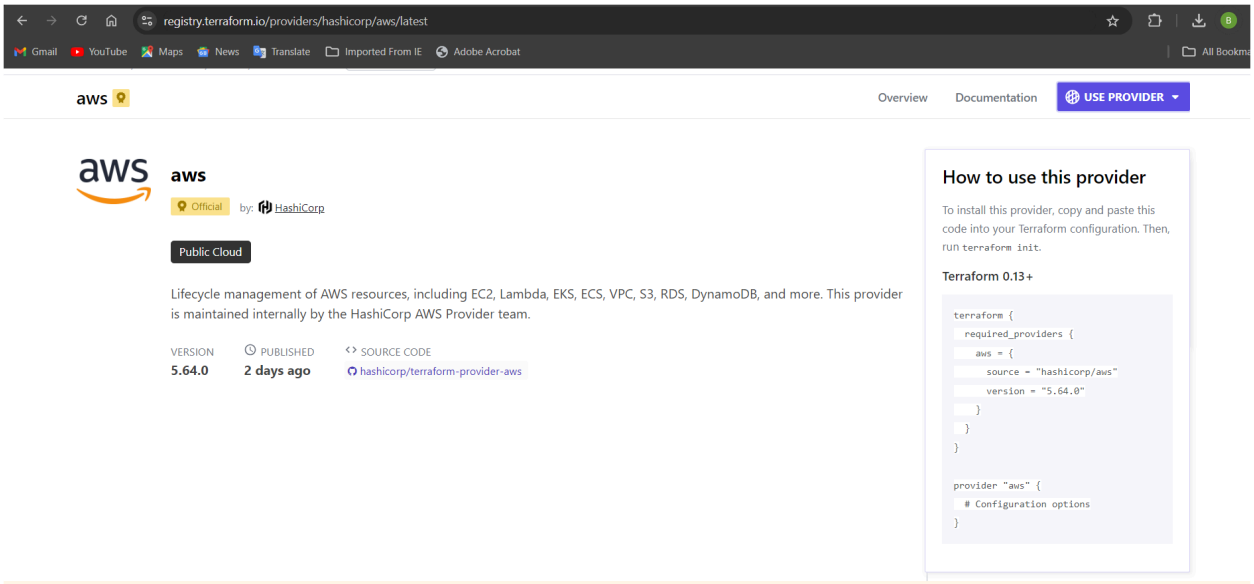
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 5: In provider.tf add the following script and do not forget to add your key values and region inside the “ ” quotes all details you will get in AWS details in lab .

```
provider "aws" {
access_key=""
secret_key=""
token =""
region=""
}
```

Step 6 : Visit the website Terraform By Hashicrop go inside registry then click on Browse provider then click on or search AWS then click on use provider and copy the code exect provider part. Because provider part we have already cover.Paste the code in main.tf.



```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.64.0"
    }
  }
}
```

Step 7: Now write the following code in main.tf to create the bucket.

```
resource "aws_s3_bucket" "any_name" {

  bucket = "Bucket_Name_It_sholud_be_Unique_and_all_in_lowercase_follow_Documentation"
  tags = {
    Name="Any_name"
  }
}
```

Crediential.txt provider.tf main.tf

S3 > main.tf > resource "aws_s3_bucket" "my-bucket-us-east-1" > bucket

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.64.0"
    }
  }
}

resource "aws_s3_bucket" "my-bucket-us-east-1" {

  bucket = "bhushan-kor-aws-bucket-terraform"
  tags = {
    Name="Sample Bucket"
  }
}
```

Step 8: Now open new terminal and cd to S3 And perform the following commands
1)terraform init
2)terraform plan
3)terraform apply

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE PORTS

```
Microsoft Windows [Version 10.0.22631.4037]
(c) Microsoft Corporation. All rights reserved.

C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts>cd S3
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.64.0"...
- Installing hashicorp/aws v5.64.0...
- Installed hashicorp/aws v5.64.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

```
C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.my-bucket-us-east-1 will be created
  + resource "aws_s3_bucket" "my-bucket-us-east-1" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "bhushan-Kor-aws-bucket-terraform"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
      + id                          = (known after apply)
      + object_lock_enabled         = (known after apply)
      + policy                      = (known after apply)
      + region                      = (known after apply)
      + request_payer               = (known after apply)
      + tags                        = {
          + "Name" = "Sample Bucket"
        }
      + tags_all                    = {
          + "Name" = "Sample Bucket"
        }
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)

      + cors_rule (known after apply)

      + grant (known after apply)

      + lifecycle_rule (known after apply)
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
      + lifecycle_rule (known after apply)

      + logging (known after apply)

      + object_lock_configuration (known after apply)

      + replication_configuration (known after apply)

      + server_side_encryption_configuration (known after apply)

      + versioning (known after apply)

      + website (known after apply)
    }

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

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
```

```
C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.my-bucket-us-east-1 will be created
  + resource "aws_s3_bucket" "my-bucket-us-east-1" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "bhushan-kor-aws-bucket-terraform"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
      + id                          = (known after apply)
      + object_lock_enabled         = (known after apply)
      + policy                      = (known after apply)
      + region                      = (known after apply)
      + request_payer               = (known after apply)
      + tags                        = {
          + "Name" = "Sample Bucket"
        }
      + tags_all                    = {
          + "Name" = "Sample Bucket"
        }
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)

      + cors_rule (known after apply)

      + grant (known after apply)
```

```
      + cors_rule (known after apply)

      + grant (known after apply)

      + lifecycle_rule (known after apply)

      + logging (known after apply)

      + object_lock_configuration (known after apply)

      + replication_configuration (known after apply)

      + server_side_encryption_configuration (known after apply)

      + versioning (known after apply)

      + website (known after apply)
      + website (known after apply)
    }
      + website (known after apply)
    }

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

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

  Enter a value: yes

aws_s3_bucket.my-bucket-us-east-1: Creating...
aws_s3_bucket.my-bucket-us-east-1: Creation complete after 8s [id=bhushan-kor-aws-bucket-terraform]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

      + website (known after apply)
    }
```

```
       + website (known after apply)
     }

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

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

   Enter a value: yes

 aws_s3_bucket.my-bucket-us-east-1: Creating...
 aws_s3_bucket.my-bucket-us-east-1: Creation complete after 8s [id=bhushan-kor-aws-bucket-terraform]

 Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

 C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>
       + website (known after apply)
     }

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

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

   Enter a value: yes

 aws_s3_bucket.my-bucket-us-east-1: Creating...
 aws_s3_bucket.my-bucket-us-east-1: Creation complete after 8s [id=bhushan-kor-aws-bucket-terraform]

 Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

 C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>


       + website (known after apply)
```

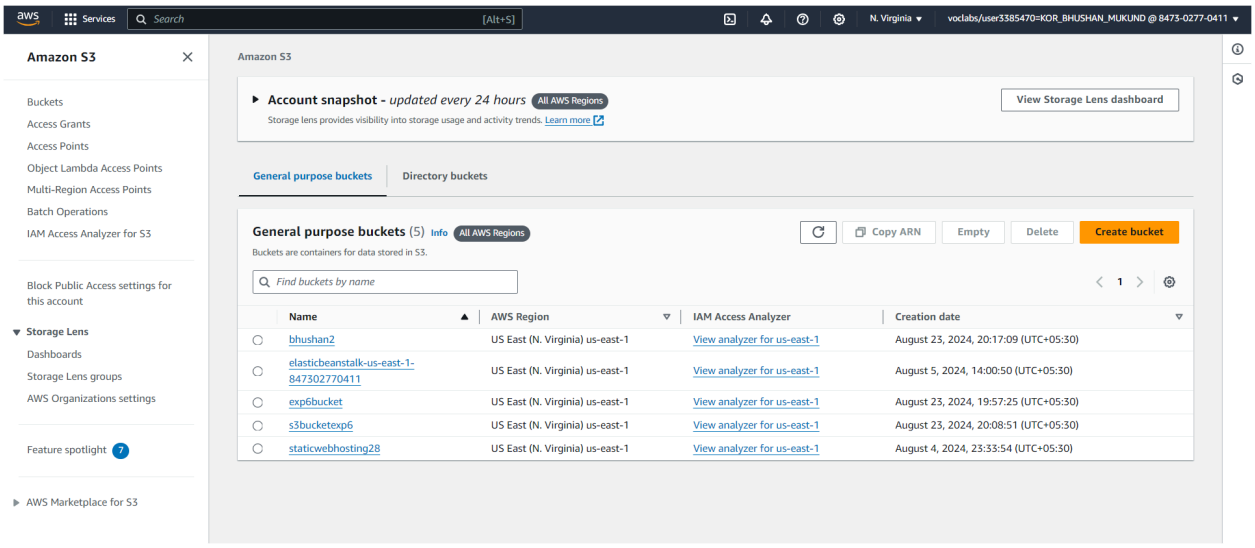
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
      + website (known after apply)
    }

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

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

  Enter a value: yes

aws_s3_bucket.my-bucket-us-east-1: Creating...
aws_s3_bucket.my-bucket-us-east-1: Creation complete after 8s [id=bhushan-kor-aws-bucket-terraform]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>



    }

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

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

  Enter a value: yes

aws_s3_bucket.my-bucket-us-east-1: Creating...
aws_s3_bucket.my-bucket-us-east-1: Creation complete after 8s [id=bhushan-kor-aws-bucket-terraform]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>
```

Now If you Click on AWS on lab it will open your AWS account and in S3 you can see you bucket.

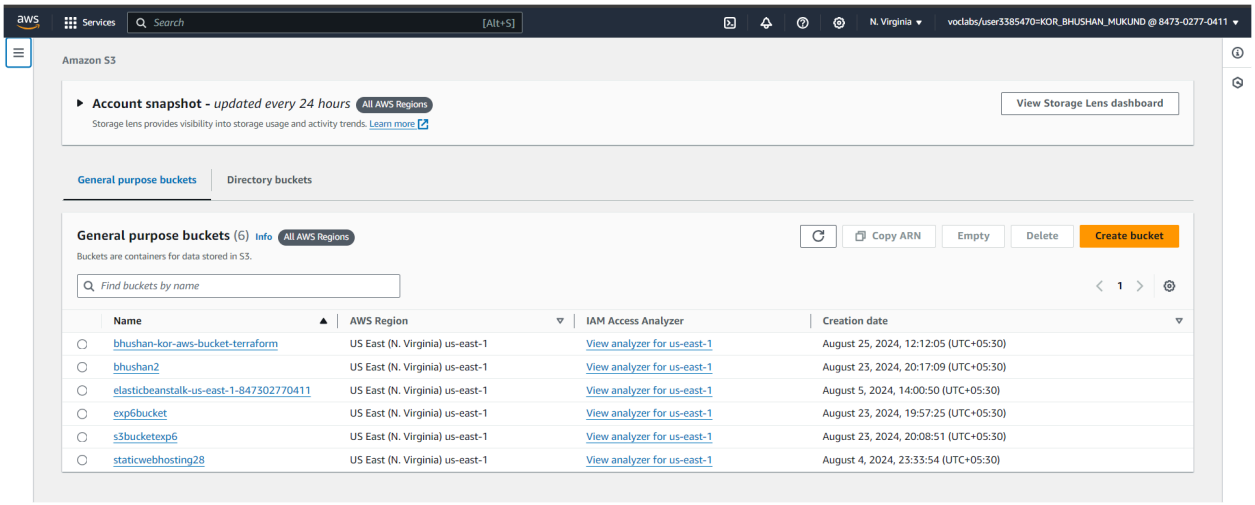
Before terrafrom Apply:



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

After terraform apply:



Step 9: Now to destroy the bucket run command terraform destroy.

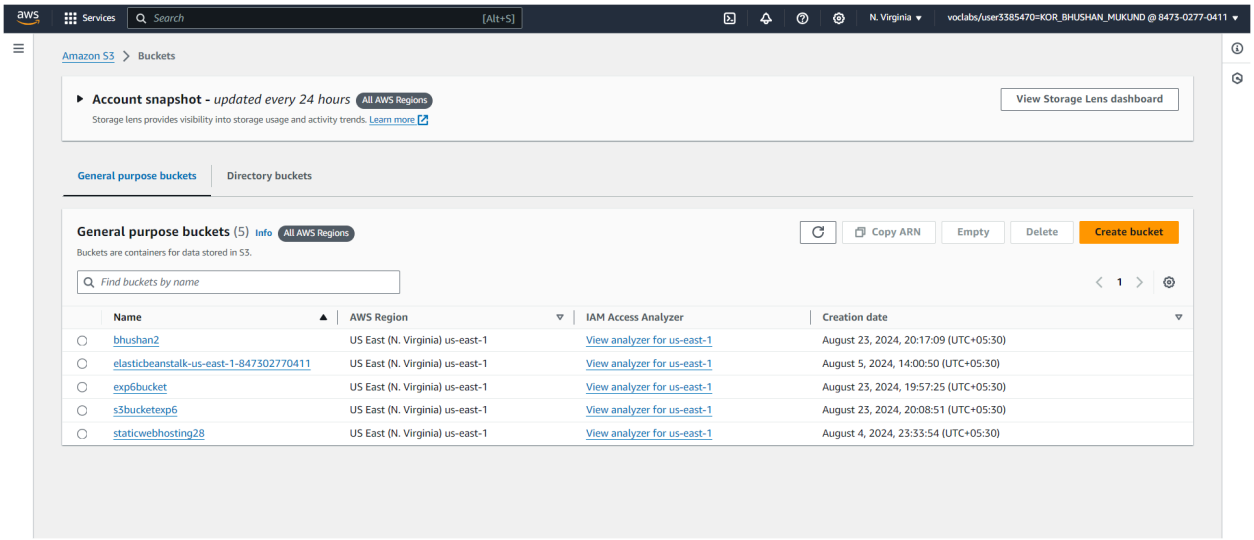
```
C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>terraform destroy
aws_s3_bucket.my-bucket-us-east-1: Refreshing state... [id=bhushan-kor-aws-bucket-terraform]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_s3_bucket.my-bucket-us-east-1 will be destroyed
  - resource "aws_s3_bucket" "my-bucket-us-east-1" {
      - arn                         = "arn:aws:s3:::bhushan-kor-aws-bucket-terraform" -> null
      - bucket                      = "bhushan-kor-aws-bucket-terraform" -> null
      - bucket_domain_name          = "bhushan-kor-aws-bucket-terraform.s3.amazonaws.com" -> null
      - bucket_regional_domain_name = "bhushan-kor-aws-bucket-terraform.s3.us-east-1.amazonaws.com" -> null
      - force_destroy               = false -> null
      - hosted_zone_id              = "Z3AQBSTGFYJSTF" -> null
      - id                          = "bhushan-kor-aws-bucket-terraform" -> null
      - object_lock_enabled         = false -> null
      - region                      = "us-east-1" -> null
      - request_payer               = "BucketOwner" -> null
      - tags                        = {
          - "Name" = "Sample Bucket"
        } -> null
      - tags_all                    = {
          - "Name" = "Sample Bucket"
        } -> null
        # (3 unchanged attributes hidden)

      - grant {
          - id          = "778043bd0e67860760caebd7a1a61d745d8798fa35ab31144e54d7003ee08ae8" -> null
          - permissions = [
              - "FULL_CONTROL",
            ] -> null
          - type        = "CanonicalUser" -> null
            # (1 unchanged attribute hidden)
        }
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
            - "FULL_CONTROL",
          ] -> null
        - type         = "CanonicalUser" -> null
          # (1 unchanged attribute hidden)
      }

    - server_side_encryption_configuration {
        - rule {
            - bucket_key_enabled = false -> null

            - apply_server_side_encryption_by_default {
                - sse_algorithm     = "AES256" -> null
                  # (1 unchanged attribute hidden)
              }
          }
      }

    - versioning {
        - enabled    = false -> null
        - mfa_delete = false -> null
      }
  }

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

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_s3_bucket.my-bucket-us-east-1: Destroying... [id=bhushan-kor-aws-bucket-terraform]
aws_s3_bucket.my-bucket-us-east-1: Destruction complete after 2s

Destroy complete! Resources: 1 destroyed.

C:\Users\bhush\one drive 2\OneDrive\Desktop\Terraform Scripts\S3>
```

After terraform destroy :



Step 10: Congratulations we are done with creating and destroying the S3 bucket on AWS using Terraform.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

Theory :

1. Static Application Security Testing (SAST)

- **Definition**: SAST is a method of debugging by examining source code before a program is run. It identifies vulnerabilities early in the development lifecycle.
- **Key Features**:
  - **Early Detection**: Finds vulnerabilities during the coding phase, reducing remediation costs.
  - **Code Quality Analysis**: Beyond security, it also assesses code quality, maintainability, and adherence to coding standards.
  - **Integration**: Can be integrated into CI/CD pipelines for continuous security assessment.

2. SonarQube

- **Definition:** SonarQube is an open-source platform for continuous inspection of code quality, which includes detecting bugs, vulnerabilities, and code smells.
- **SAST Integration:** Supports SAST tools to analyze code and provide metrics and reports within the SonarQube dashboard.
- **Quality Gates:** Allows setting thresholds (quality gates) that must be met before code can proceed in the CI/CD pipeline.

3. Reporting and Remediation

- **Results Analysis**: After SAST scans, results are usually presented in a detailed report highlighting vulnerabilities, their severity, and remediation advice.
- **Feedback Loop**: Integrating SAST results into the development workflow helps create a feedback loop, encouraging developers to address vulnerabilities proactively.

4. Best Practices

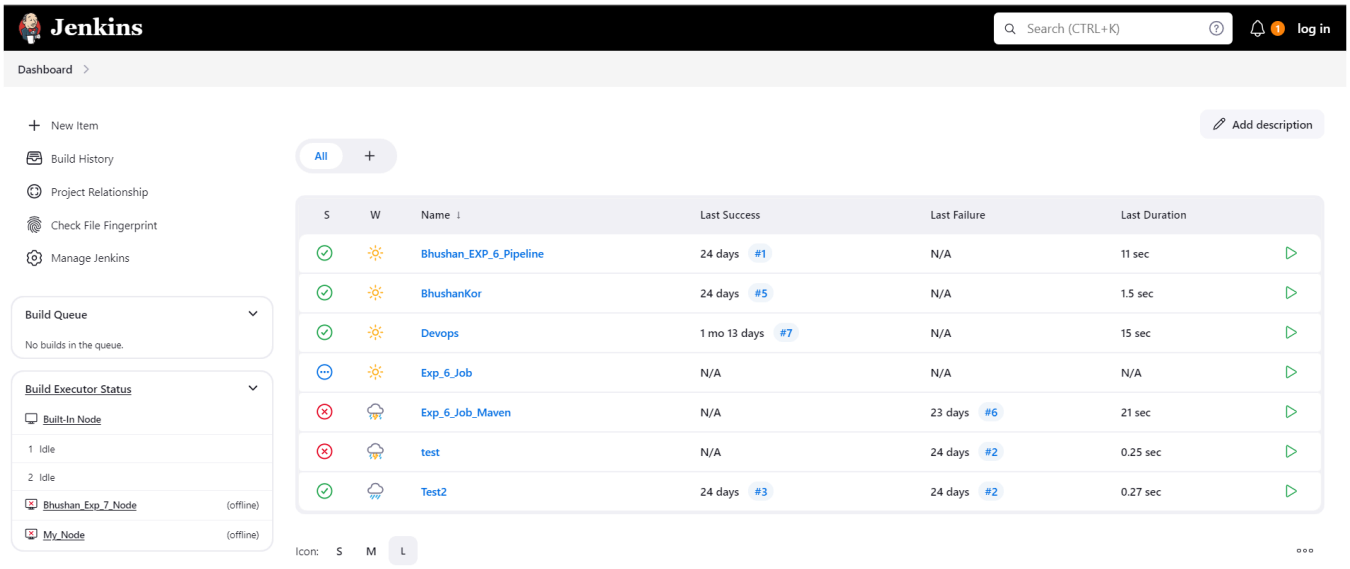
- **Regular Updates**: Keep SAST tools and configurations updated to recognize the latest vulnerabilities.
- **Customization**: Tailor SAST rules and configurations to suit the specific needs of the project and team.
- **Training**: Ensure developers are trained on security best practices to understand and mitigate vulnerabilities effectively.

Prerequisites:
- Jenkins installed
- Docker Installed (for SonarQube)

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 1: Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



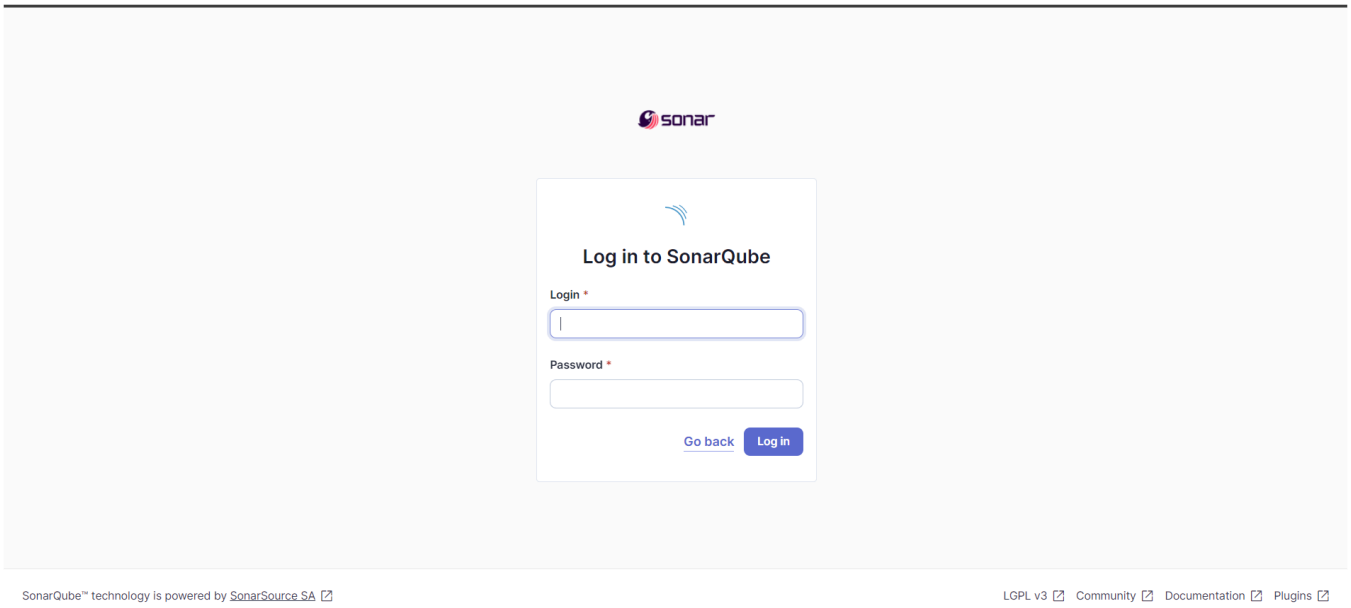
Step 2: Run SonarQube in a Docker container using this command
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
C:\Users\bhush\one drive 2\OneDrive\Desktop\Docker>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=t
rue -p 9000:9000 sonarqube:latest
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
762bedf4b1b7: Pull complete
95f9bd9906fa: Pull complete
a32d681e6b99: Pull complete
aabdd0a18314: Pull complete
5161e45ecd8d: Pull complete
aeb0020dfa06: Pull complete
01548d361aea: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:bb444c58c1e04d8a147a3bb12af941c57e0100a5b21d10e599384d59bed36c86
Status: Downloaded newer image for sonarqube:latest
dc00d2f31a229b2076f15c273ad750e45b74f871dd53b6d177b6f860d4fc8f0e

C:\Users\bhush\one drive 2\OneDrive\Desktop\Docker>
```

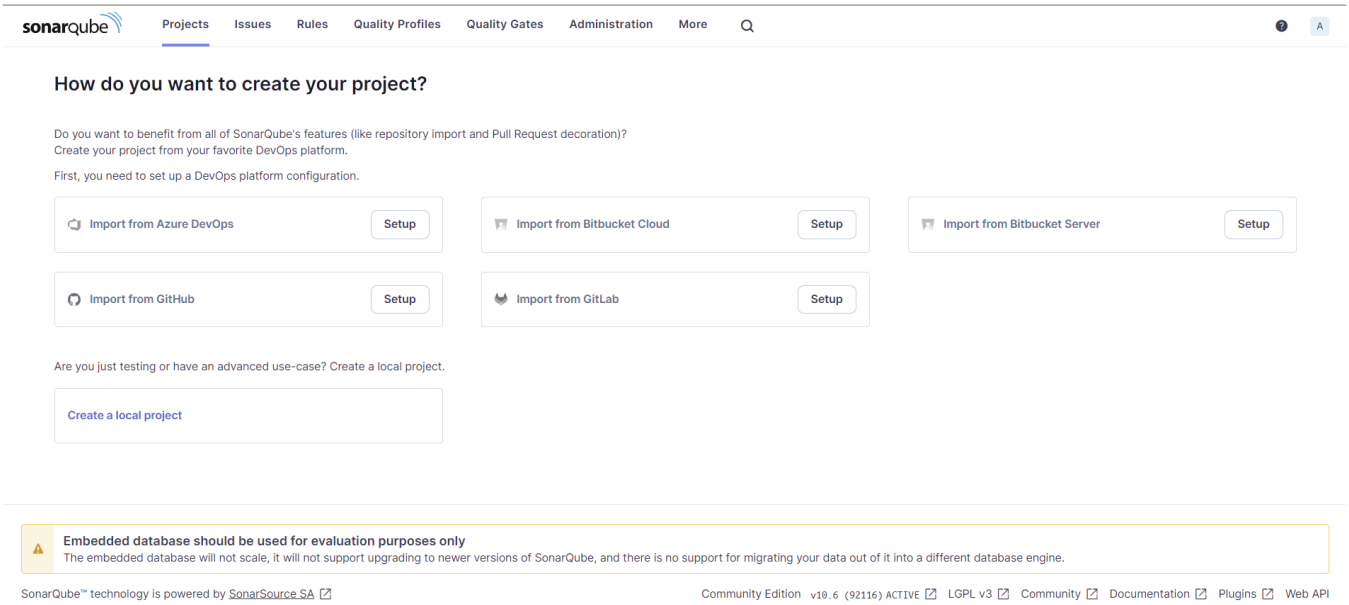
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 3: Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



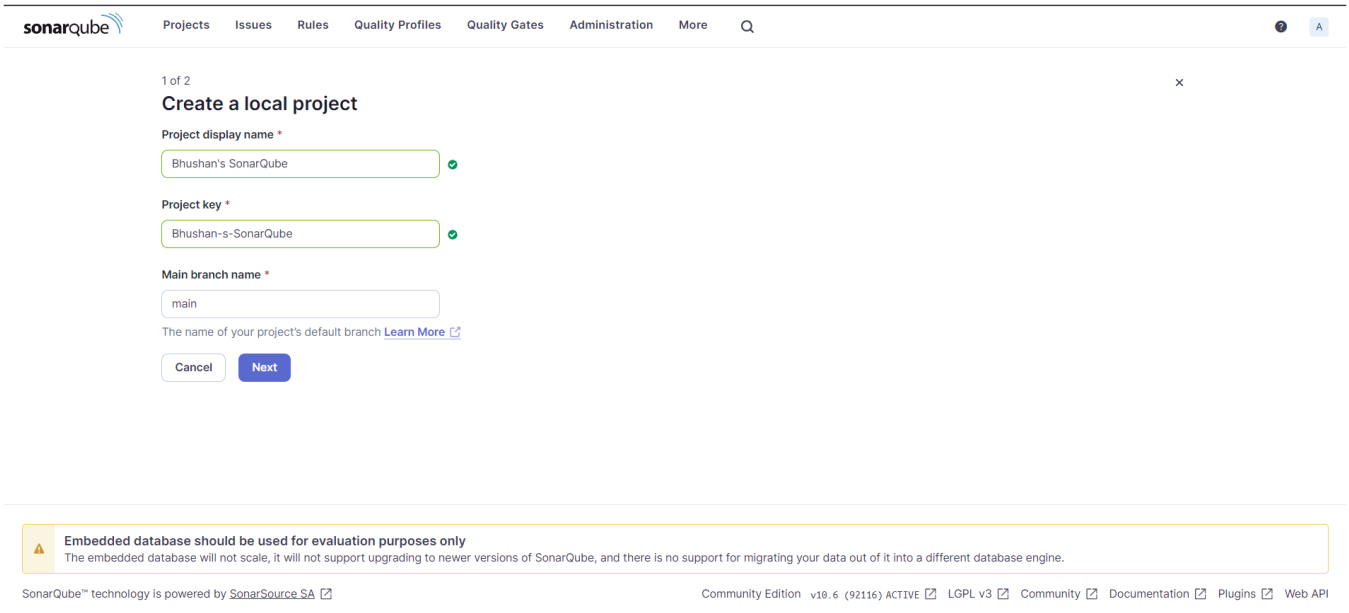
Start 4: Login to SonarQube using username admin and password admin.

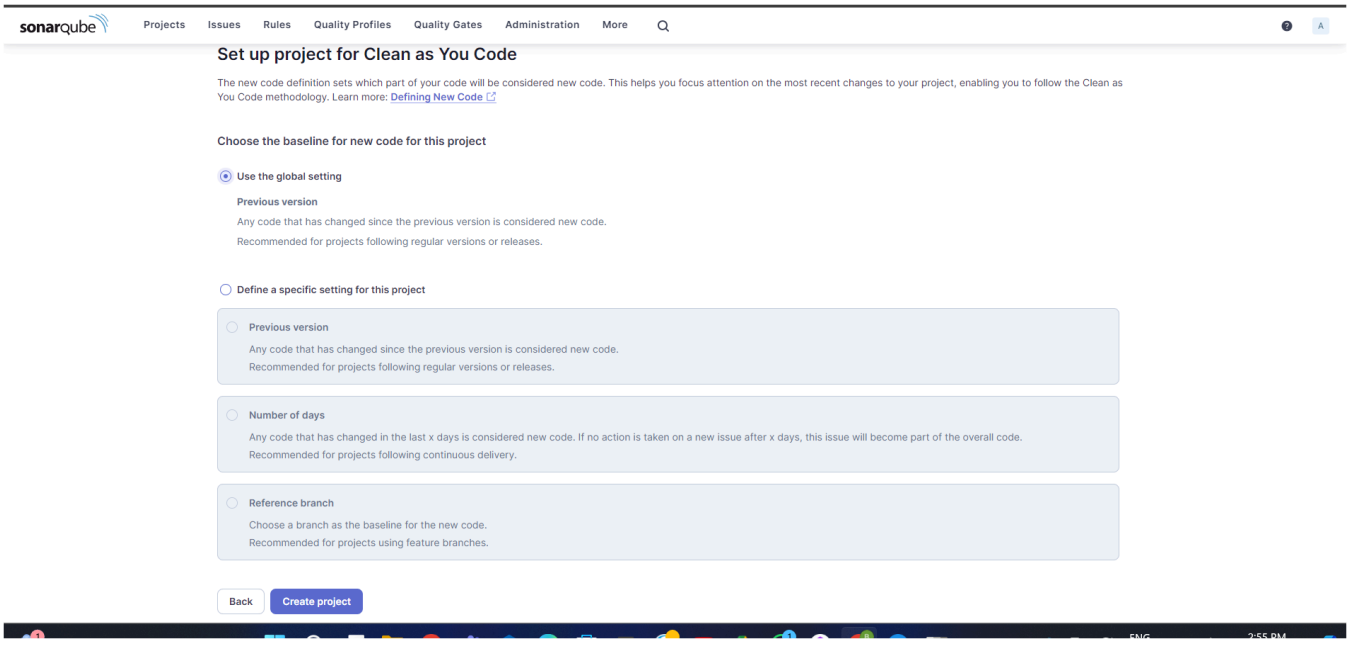


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 5: Create a manual project in SonarQube with any Name



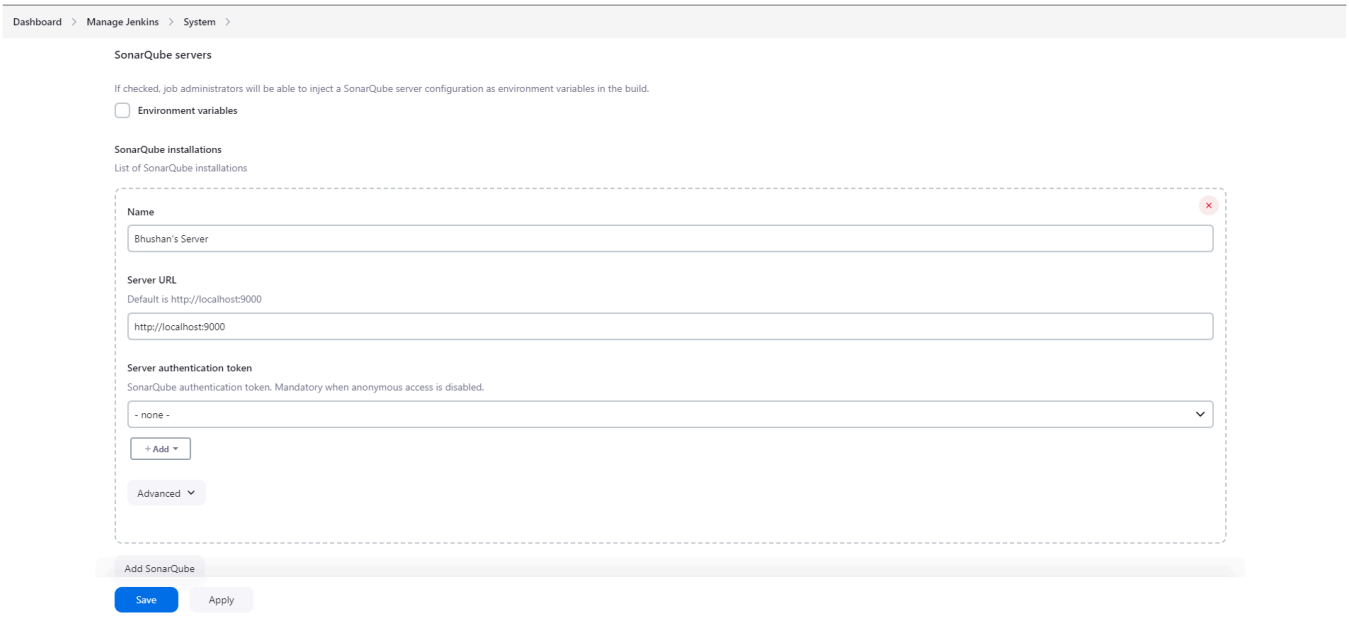


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

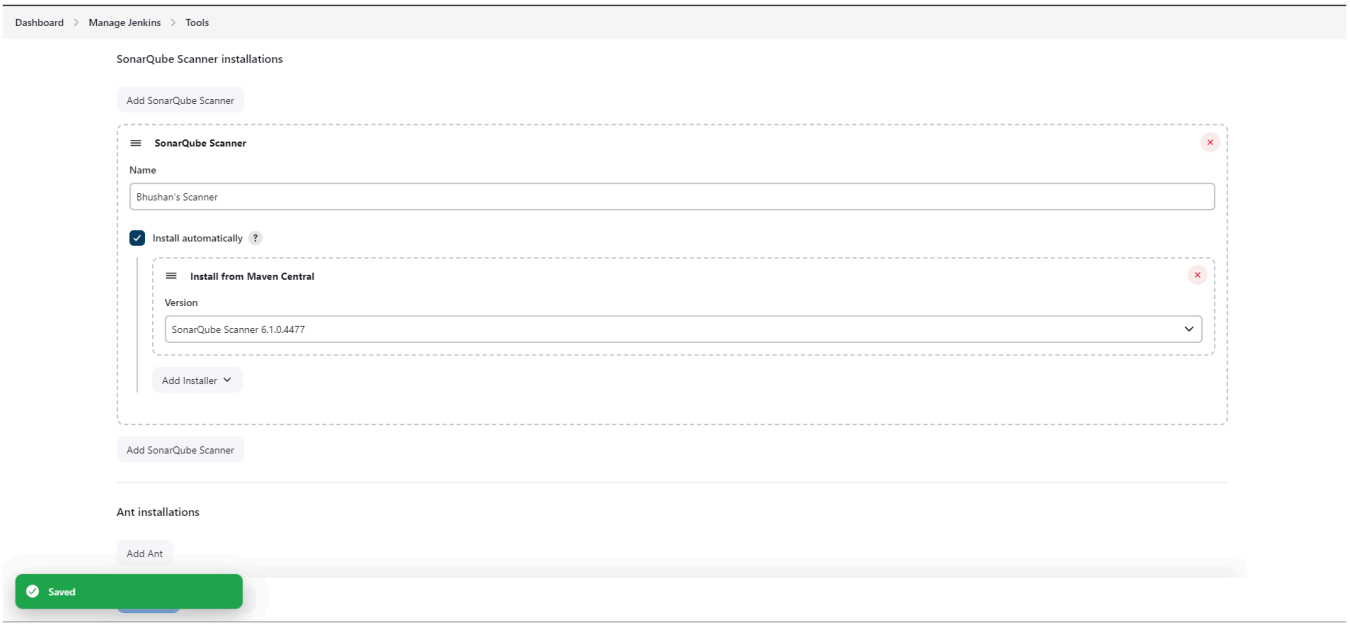
Division: D15C **Roll No: 28**

Step 6: Setup the project and come back to Jenkins Dashboard.
Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

Step 7: Under Jenkins ‘Configure System’, look for SonarQube Servers and enter the details. Enter the Server Authentication token if needed.



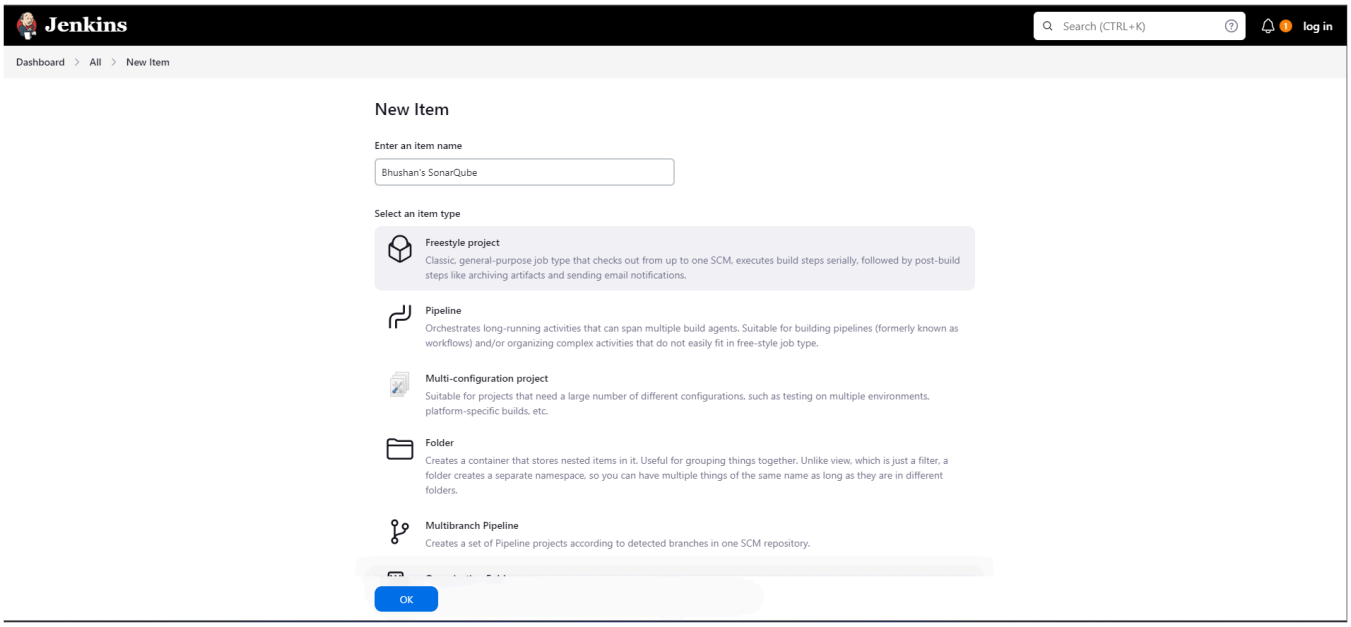
Step 8: Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.



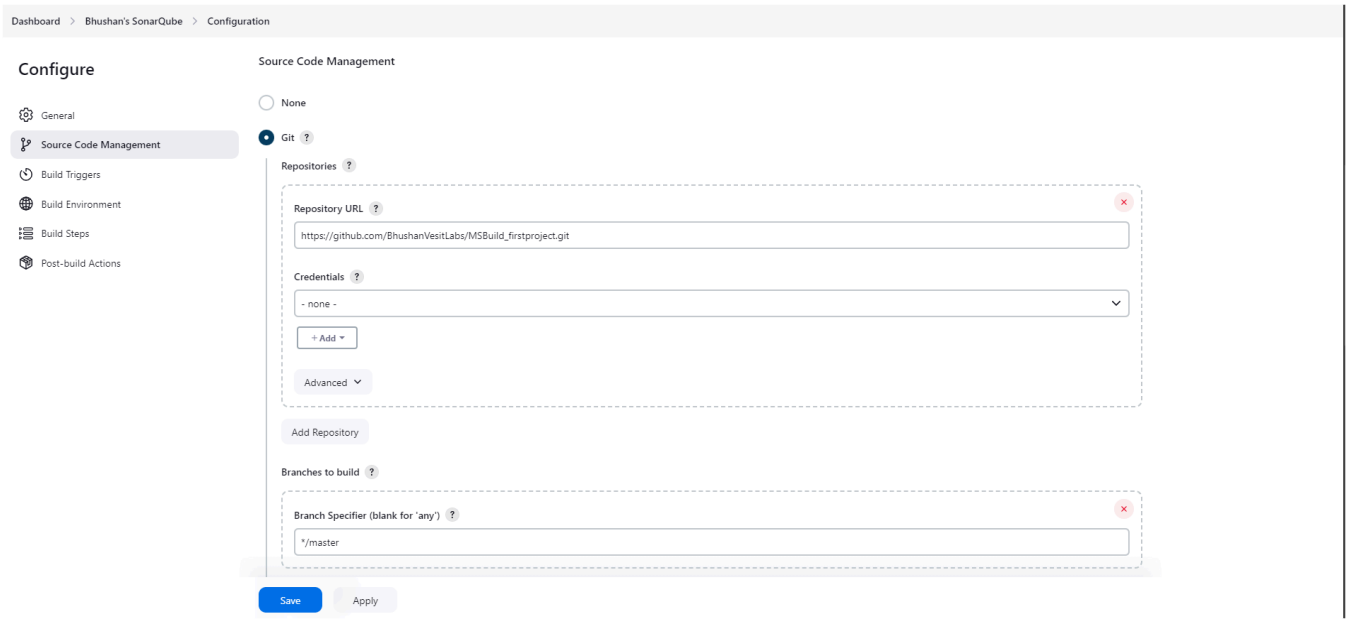
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 9: After the configuration, create a New Item in Jenkins, and choose a freestyle project.



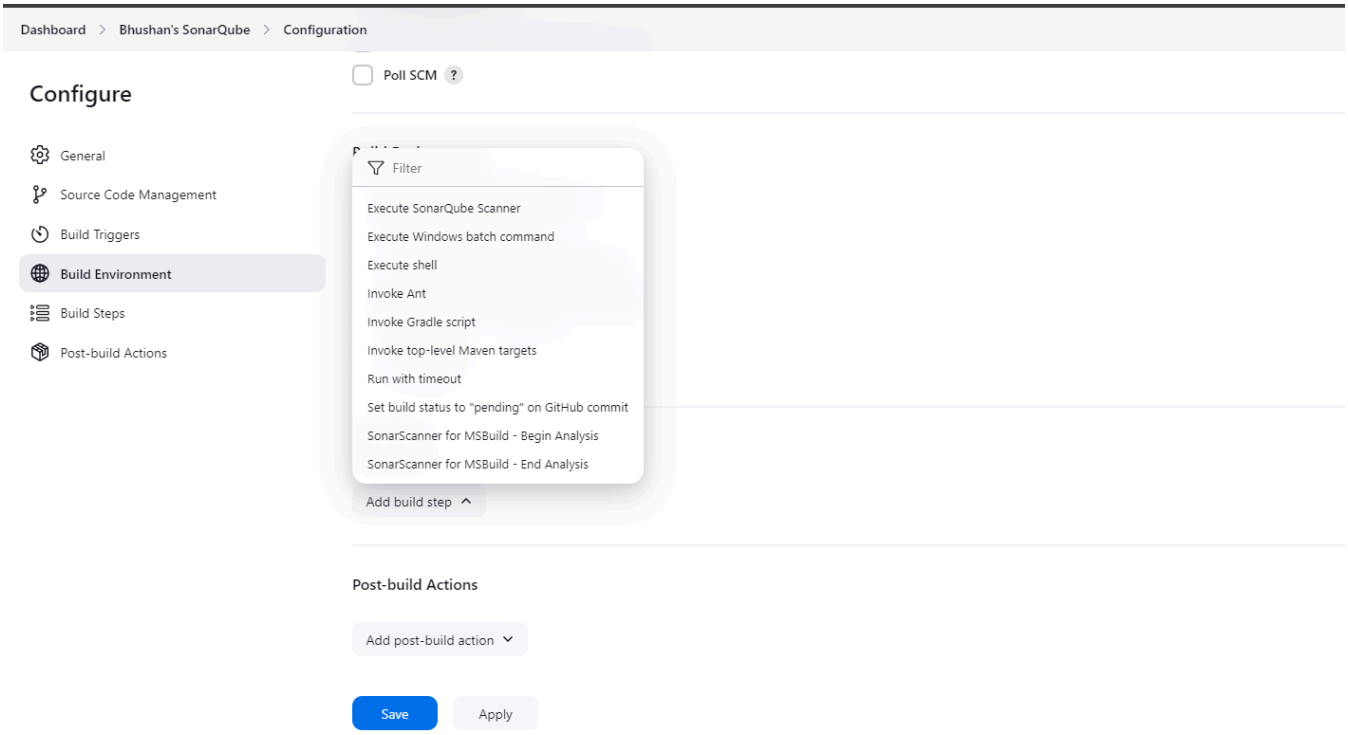
Step 10: Choose this GitHub repository in Source Code Management. https://github.com/shazforiot/MSBuild_firstproject.git .It is a sample hello-world project with no vulnerabilities and issues, just to test integration.

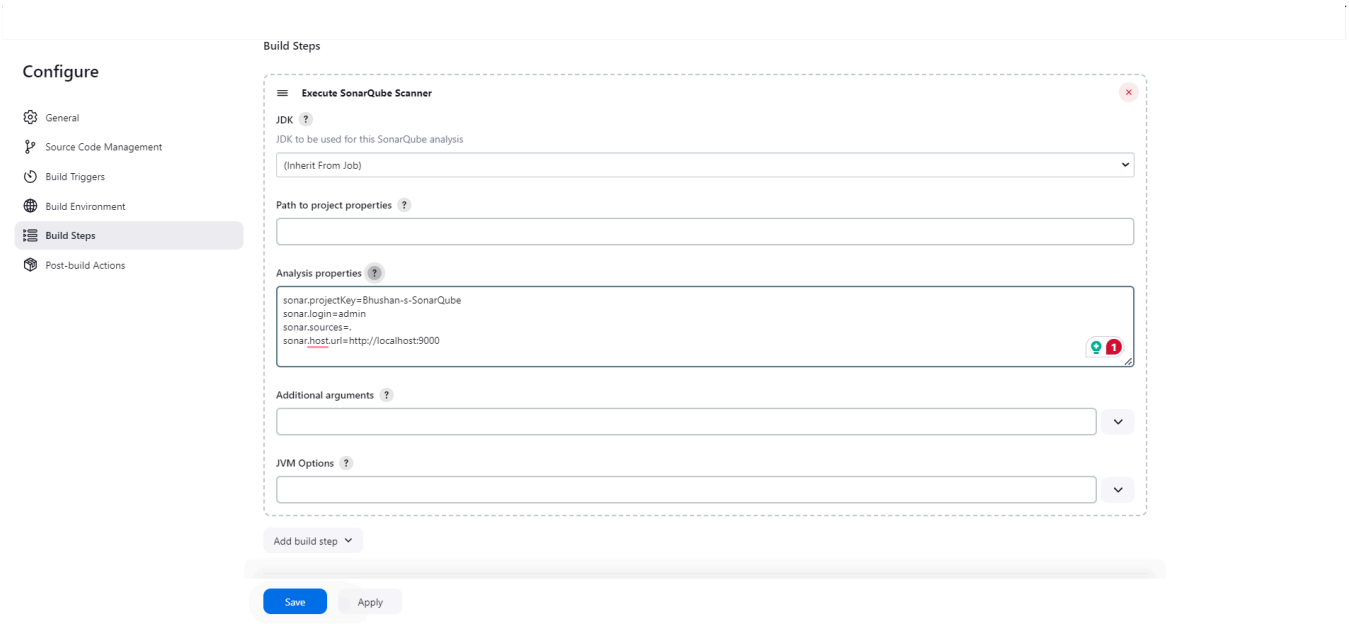


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 11: Under Build select Execute SonarQube Scanner, enter these Analysis properties. Mention the SonarQube Project Key, Login, Password, Source path and Host URL.



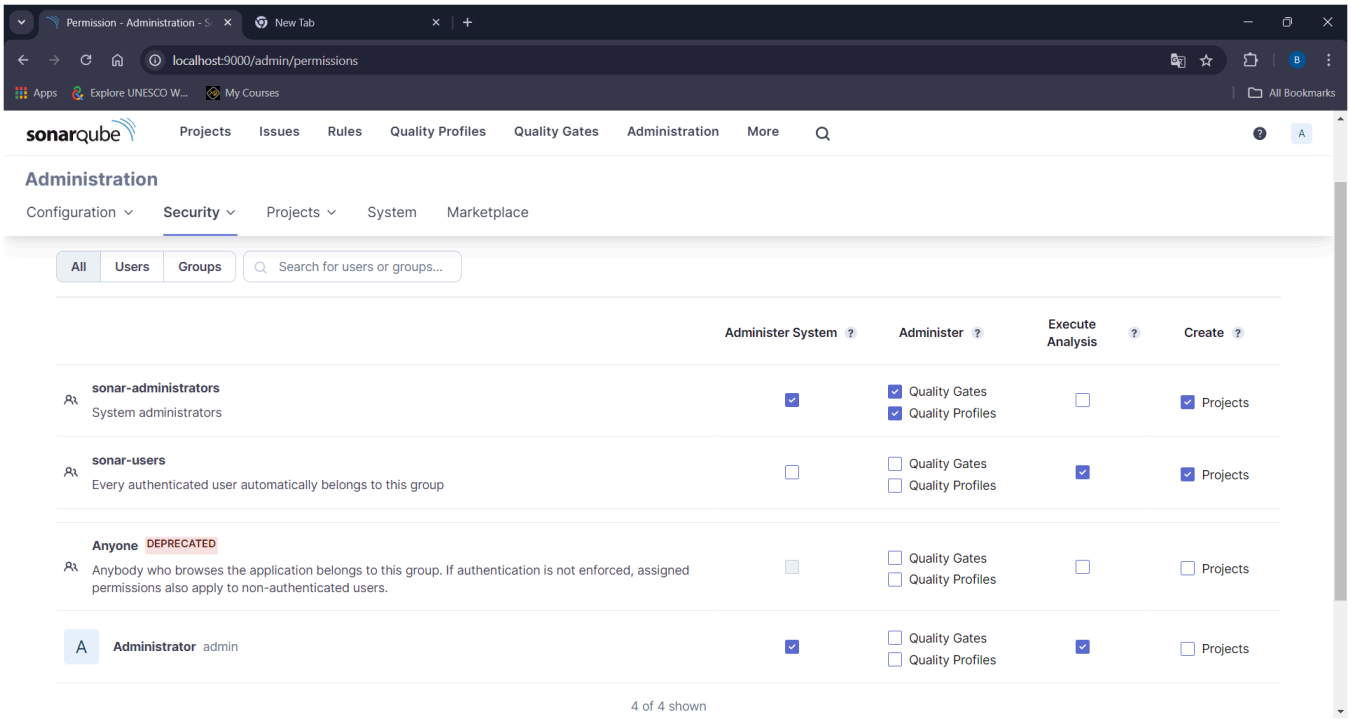


sonar.projectKey=<Your Project Key>
sonar.login=<User Name>
sonar.password=<Password>
sonar.sources=.
sonar.host.url=http://localhost:9000

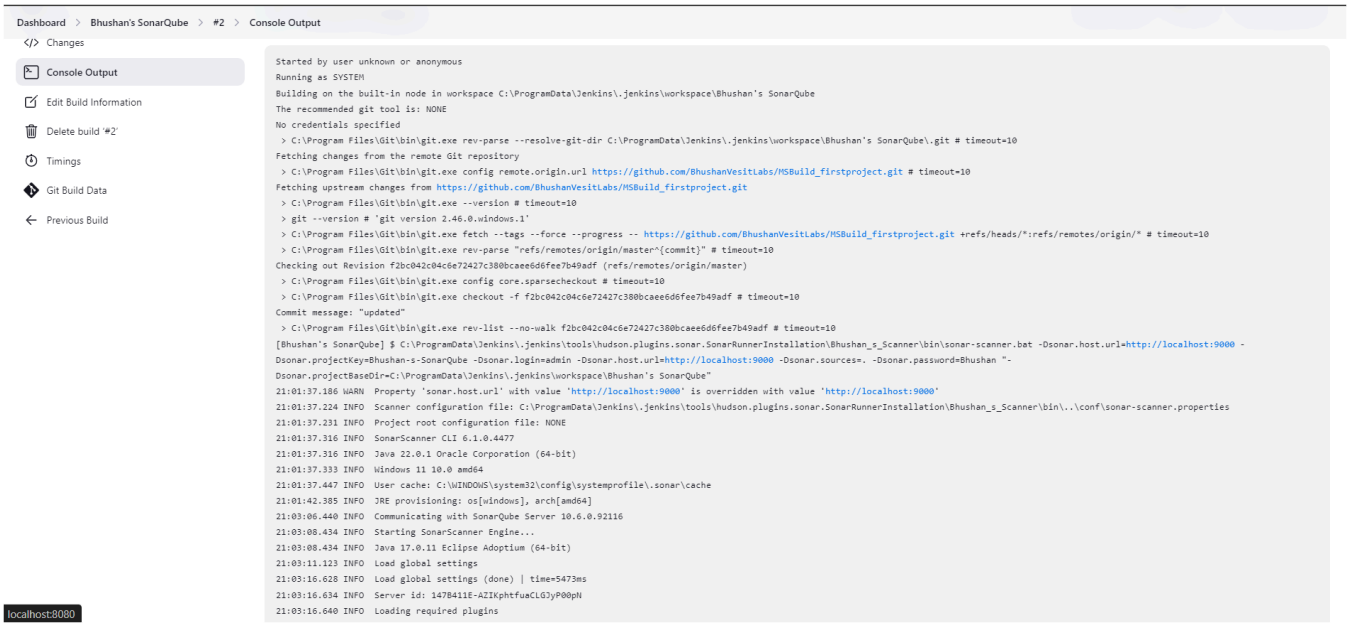
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 12: In the SonarQube go to Security then for Administrator allow Administer system and Execute analysis.

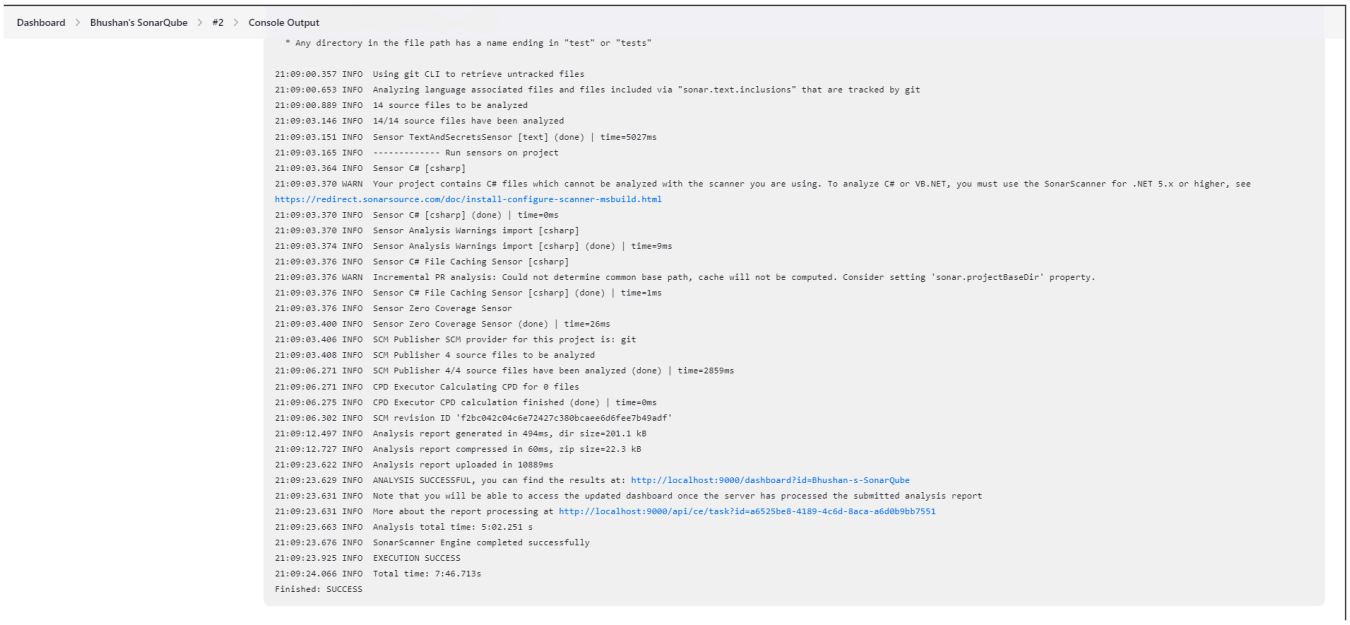


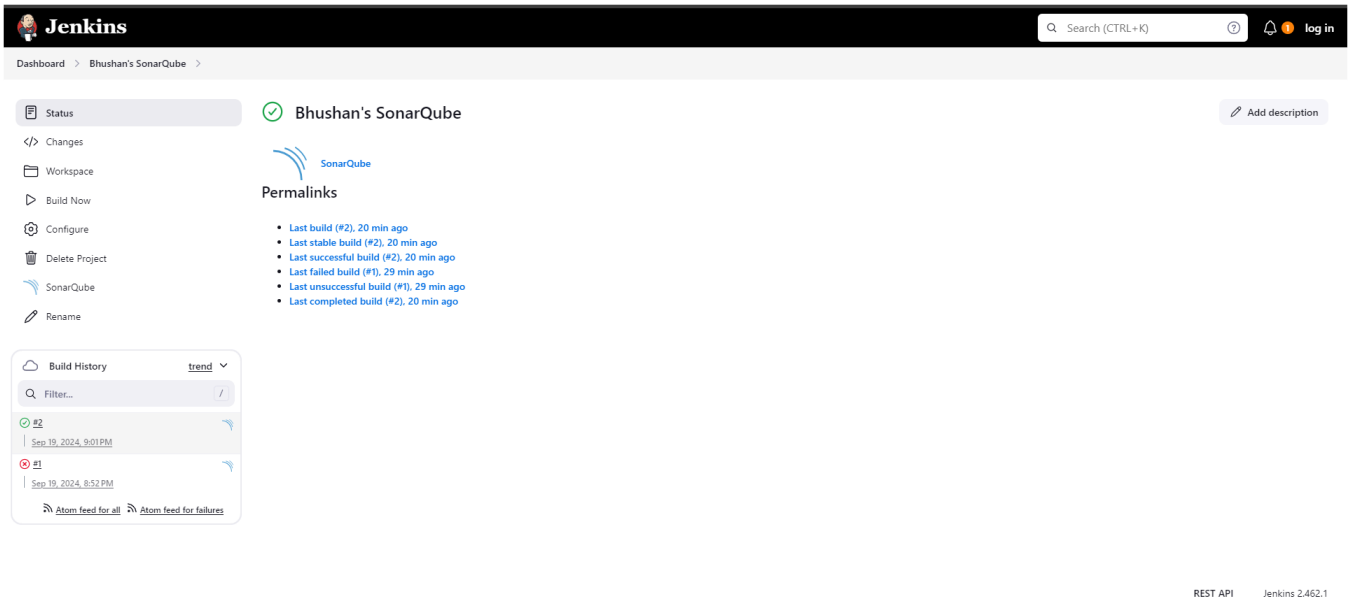
Step 13: See the Console Output.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

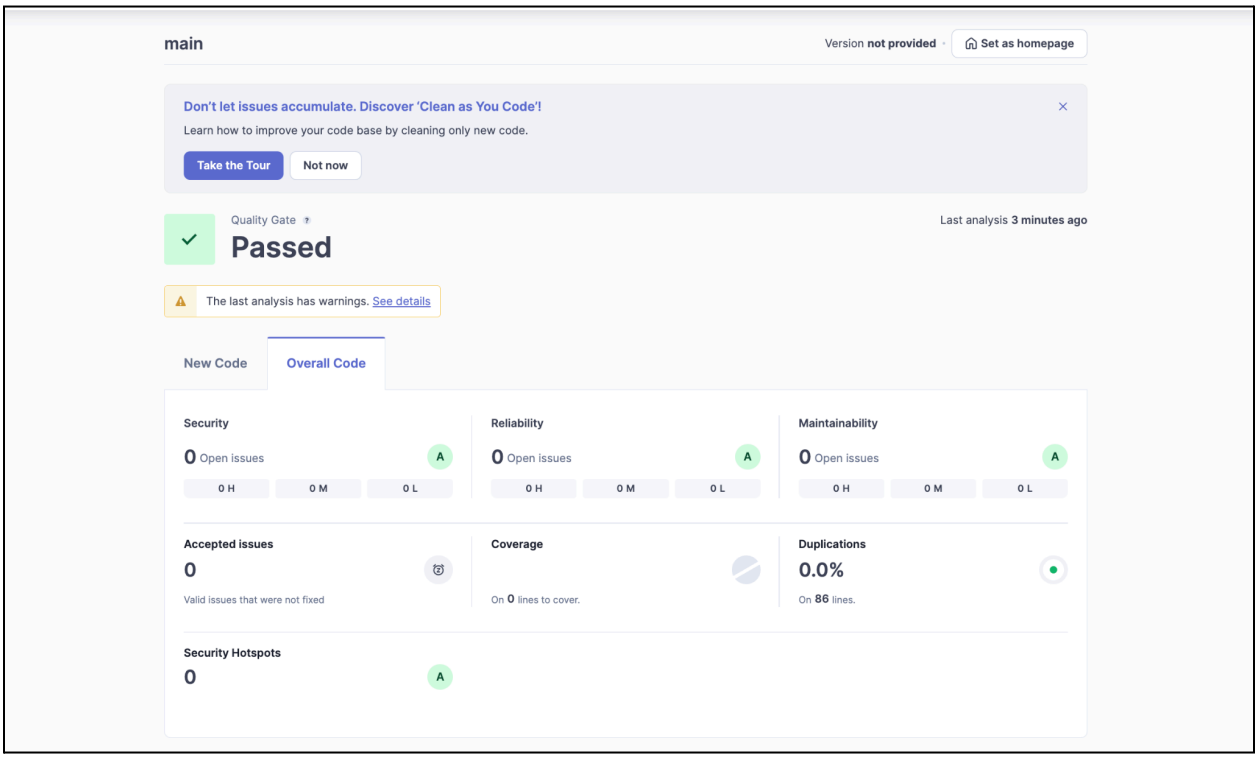




Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 14: Now See the SonarQube Project.



Conclusion :

In this experiment, we have learned how to perform Jenkins SAST using SonarQube. For this, we used a docker image of SonarQube to not install it locally on our system. After installing the required configurations on Jenkins, using a code from a GitHub repository, we analyze its code using SonarQube. Once we build the project, we can see that the SonarQube project displays that the code has no errors.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: Create a Jenkins CICD Pipeline with SonarQube / GitLab Integration to perform a static analysis of the code to detect bugs, code smells, and security vulnerabilities on a sample Web / Java / Python application.

Theory:

What is SAST?

Static Application Security Testing (SAST) is a methodology that analyzes source code to identify security vulnerabilities before compilation. It is often referred to as white box testing and helps developers detect issues early in the software development lifecycle (SDLC).

Problems SAST Solves

1. **Early Detection**: Identifies vulnerabilities in the initial development stages, reducing later risks.
2. **Real-Time Feedback**: Provides immediate insights, allowing developers to fix issues before moving forward.
3. **Code Navigation**: Offers visual representations of vulnerabilities for easier code understanding.
4. **Guidance on Fixes**: Suggests specific remediation steps without requiring deep security expertise.
5. **Comprehensive Coverage**: Analyzes the entire codebase quickly, outperforming manual reviews.
6. **Regular Scanning**: Ensures continuous security assessment through scheduled scans during builds or releases.

Importance of SAST

- **Resource Efficiency**: Automates code reviews, addressing the resource gap between developers and security staff.
- **Speed**: Processes millions of lines of code in minutes, identifying critical vulnerabilities.
- **Proactive Security**: Integrates security into the development process, preventing vulnerabilities from being overlooked.

What is a CI/CD Pipeline?

A **CI/CD Pipeline** refers to Continuous Integration and Continuous Delivery, automating software development tasks. It includes stages such as coding, building, testing, and deploying, ensuring each step is completed sequentially for efficient releases.

What is SonarQube?

SonarQube is an open-source platform for continuous code quality inspection. It performs static code analysis to generate reports on bugs, vulnerabilities, and code duplications across various programming languages.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Benefits of SonarQube

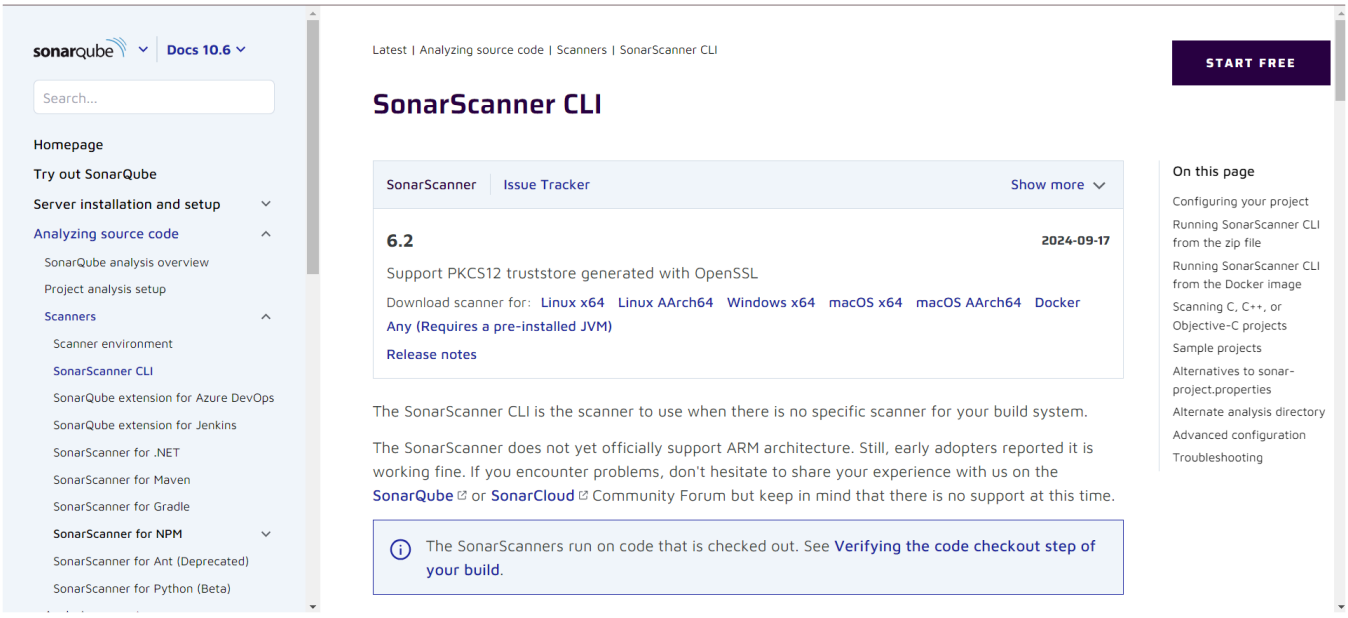
- **Sustainability**: Optimizes application lifecycle by reducing complexity and vulnerabilities.
- **Increased Productivity**: Minimizes maintenance efforts and costs.
- **Quality Control**: Integrates code quality checks into development.
- **Error Detection**: Alerts developers to fix issues before release.
- **Scalability**: Supports multiple projects without restrictions.
- **Skill Enhancement**: Provides regular feedback to improve developer skills.

Prerequisites:

- Jenkins installed
- Docker Installed (for SonarQube)
- SonarQube Docker Image

Download The SonarQube CLI according to your system :

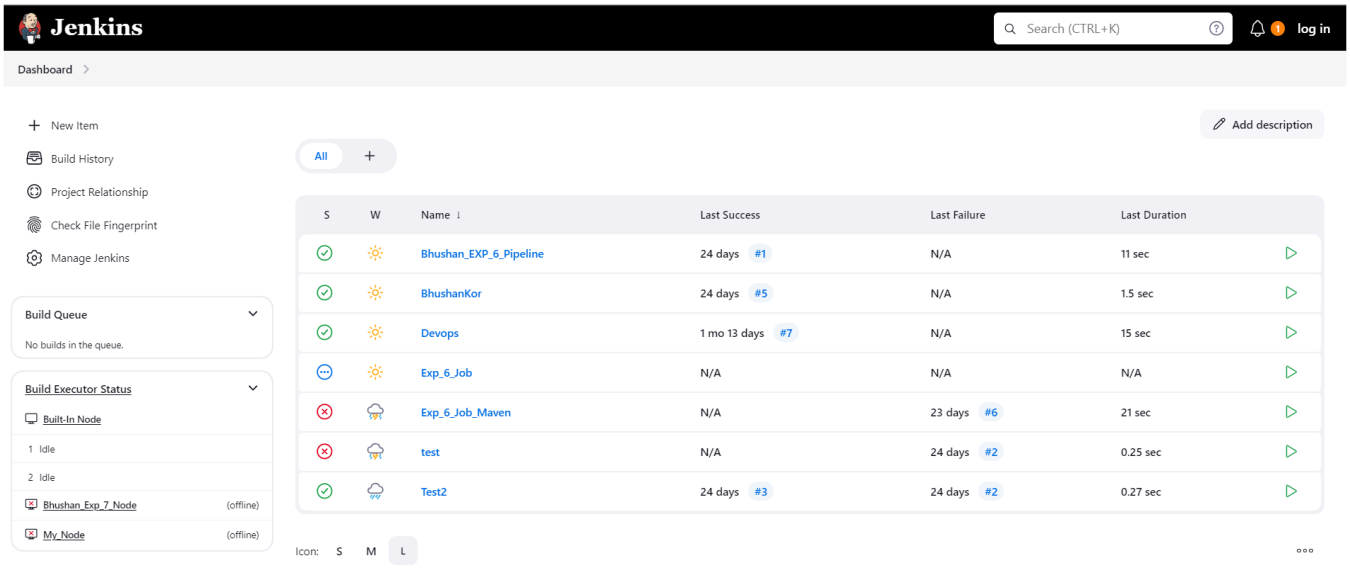
https://docs.sonarsource.com/sonarqube/latest/analyzing-source-code/scanners/sonarscanner/



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 1: Open up Jenkins Dashboard on localhost, port 8080 or whichever port it is at for you.



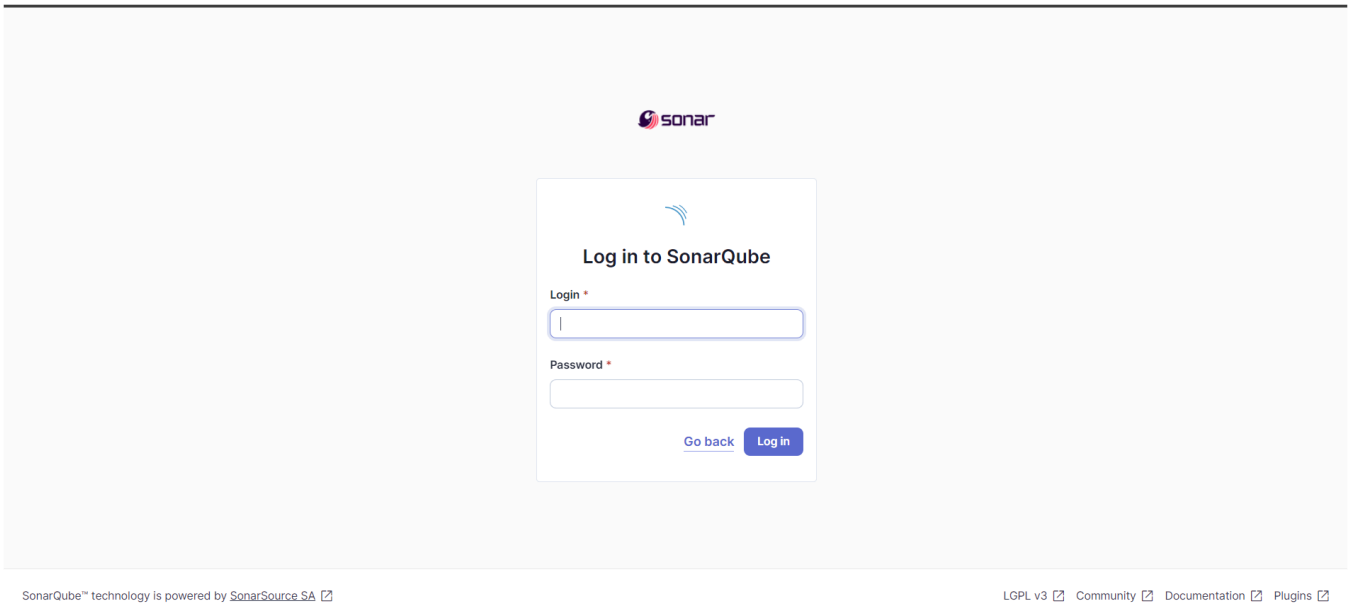
Step 2: Run SonarQube in a Docker container using this command
docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:latest

```
Microsoft Windows [Version 10.0.22621.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\INFT505-11>docker run -d --name sonarqube -e SONAR_ES_BOOTSTRAP_CHECKS_DISABLE=true -p 9000:9000 sonarqube:late
st
Unable to find image 'sonarqube:latest' locally
latest: Pulling from library/sonarqube
7478e0ac0f23: Pull complete
90a925ab929a: Pull complete
7d9a34308537: Pull complete
80338217a4ab: Pull complete
1a5fd5c7e184: Pull complete
7b87d6fa783d: Pull complete
bd819c9b5ead: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:72e9feec71242af83faf65f95a40d5e3bb2822a6c3b2cda8568790f3d31aecde
Status: Downloaded newer image for sonarqube:latest
d72b183b1866cea7ecdb976a63dfe521172c307eb45eace7b769f726f0bbf989

C:\Users\INFT505-11>
```

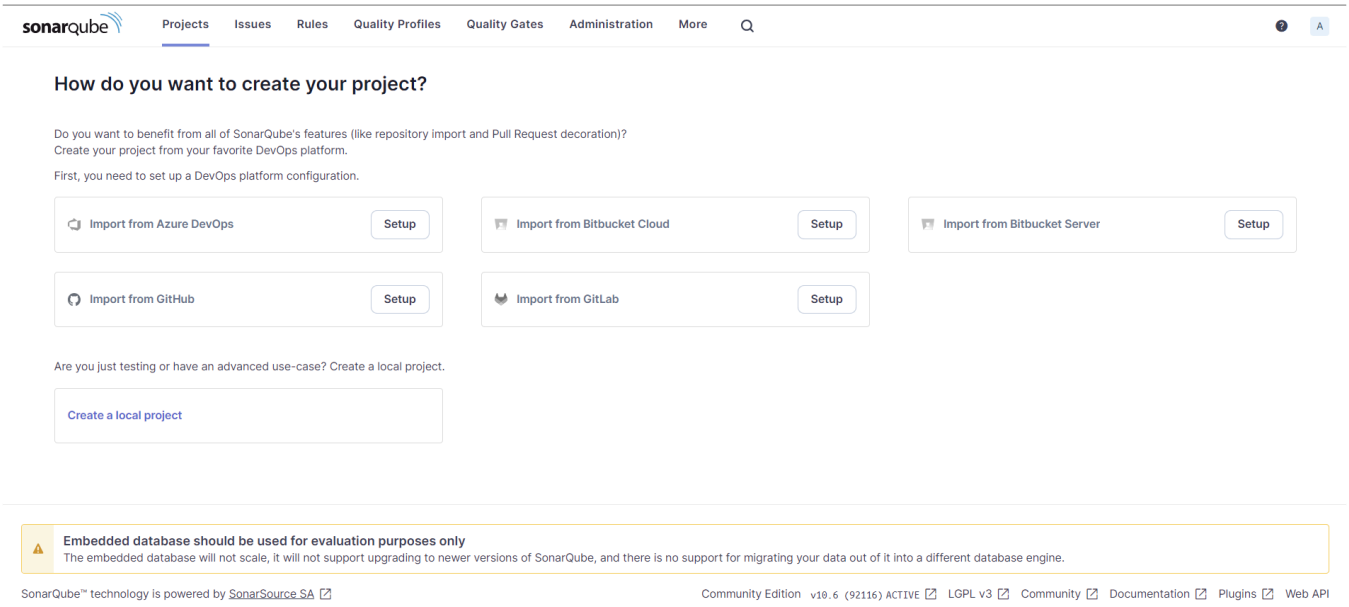
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 3: Once the container is up and running, you can check the status of SonarQube at localhost port 9000.



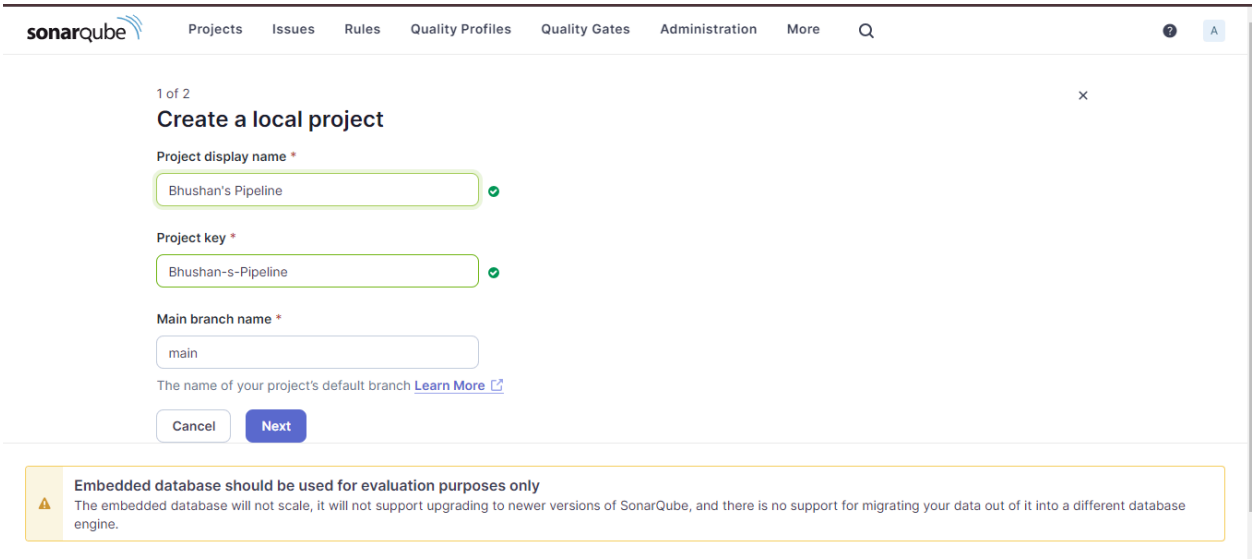
Start 4: Login to SonarQube using username admin and password admin.

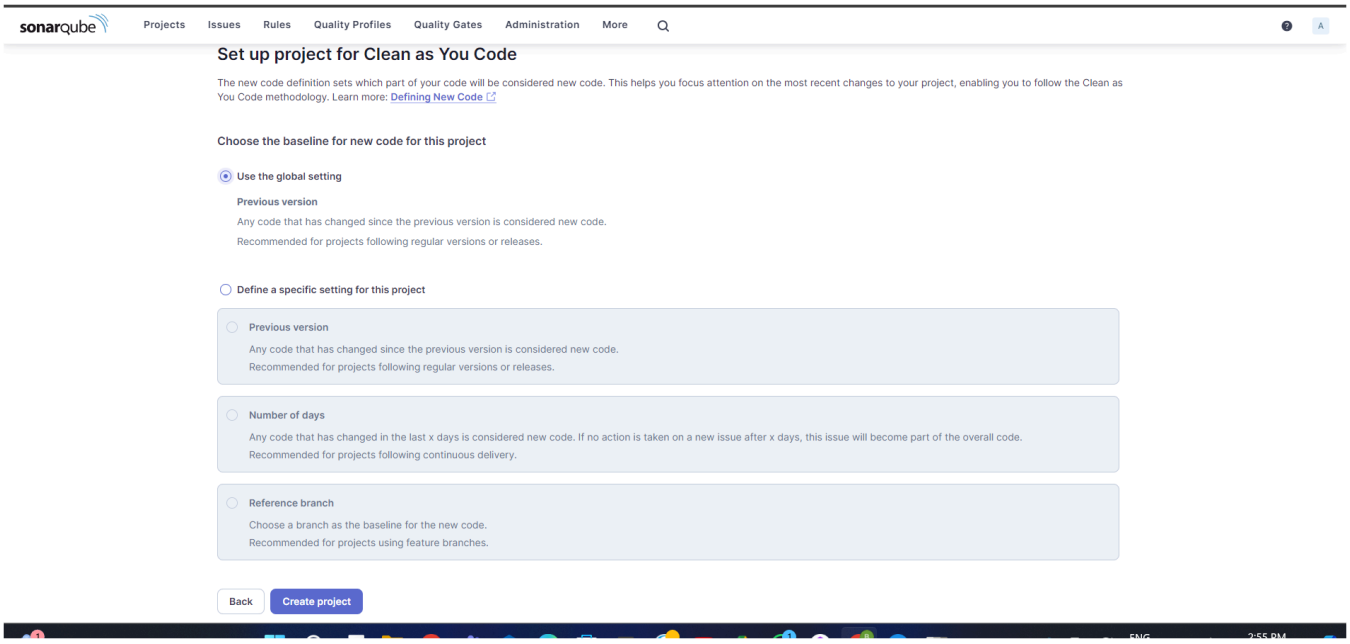


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 5: Create a manual project in SonarQube with any Name



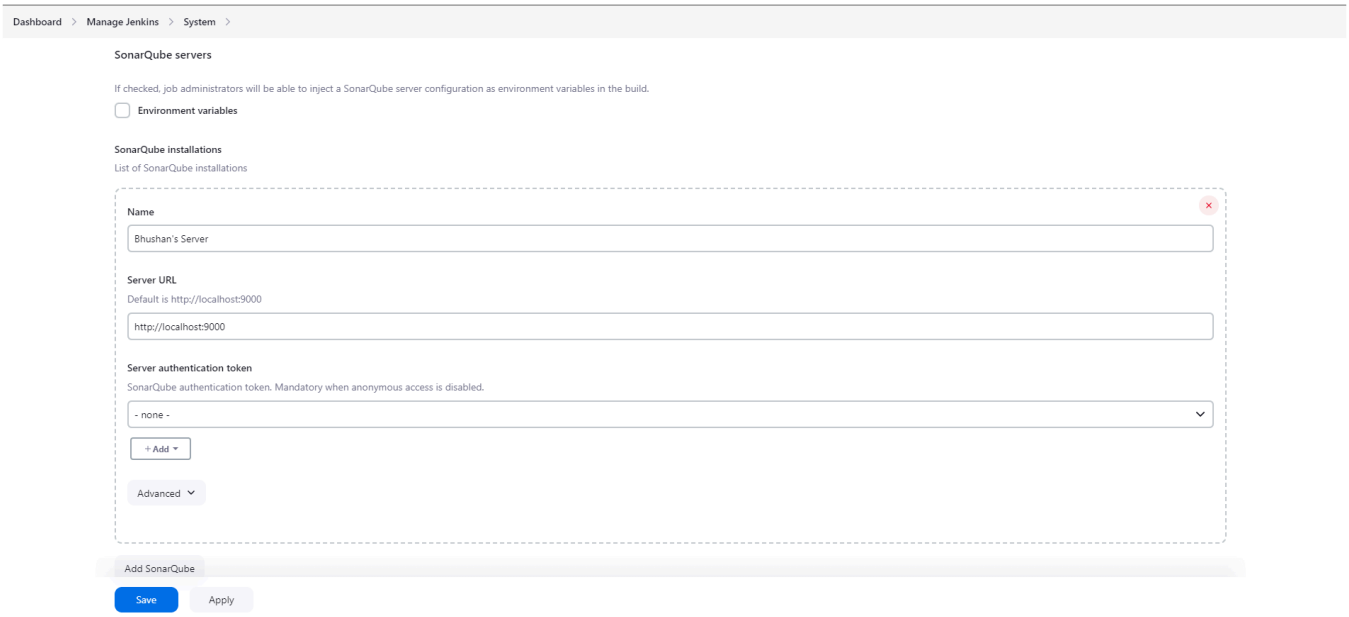


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

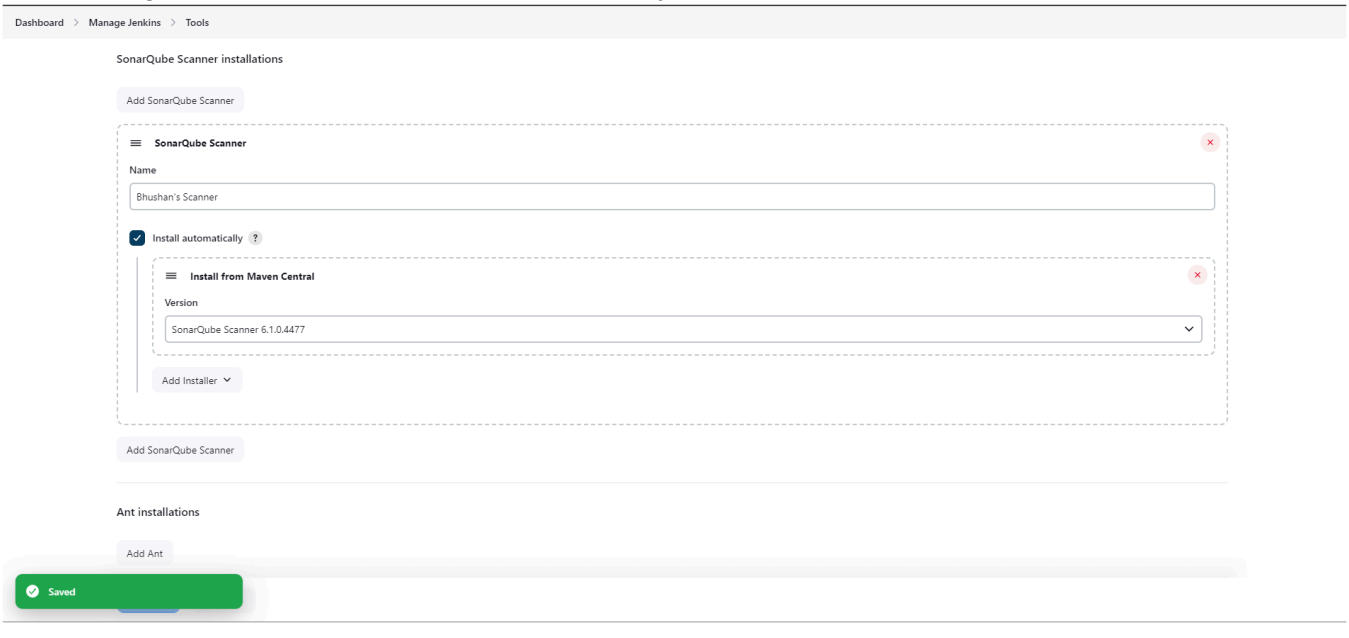
Division: D15C **Roll No: 28**

Step 6:Setup the project and come back to Jenkins Dashboard.
Go to Manage Jenkins and search for SonarQube Scanner for Jenkins and install it.

Step 7: Under Jenkins ‘Configure System’, look for SonarQube Servers and enter the details. Enter the Server Authentication token if needed.



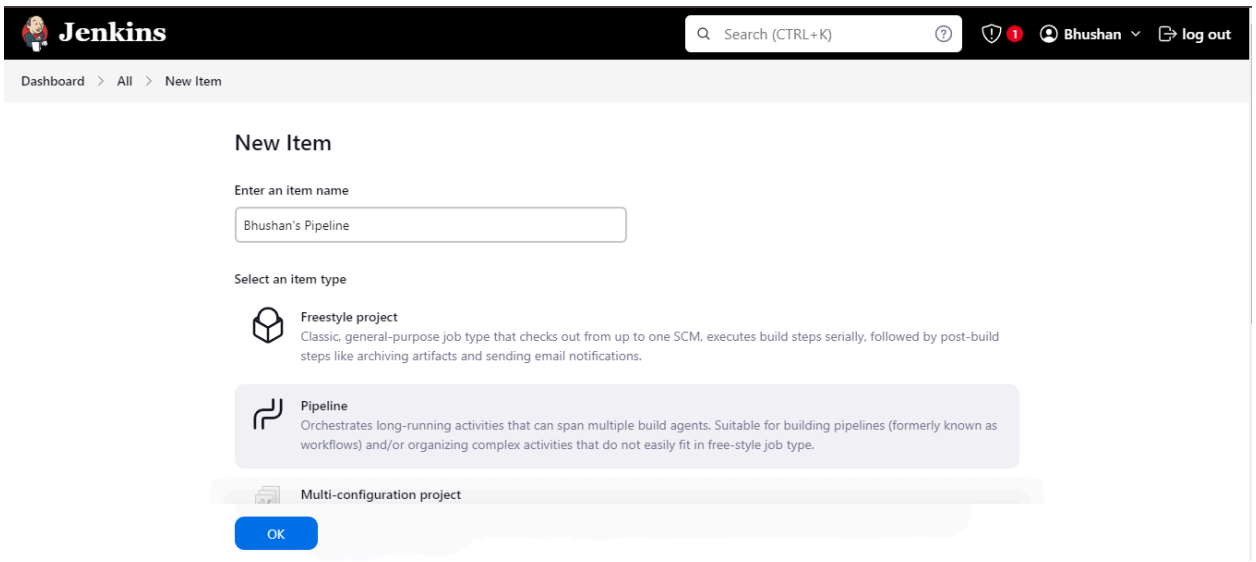
Step 8:Search for SonarQube Scanner under Global Tool Configuration. Choose the latest configuration and choose Install automatically.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

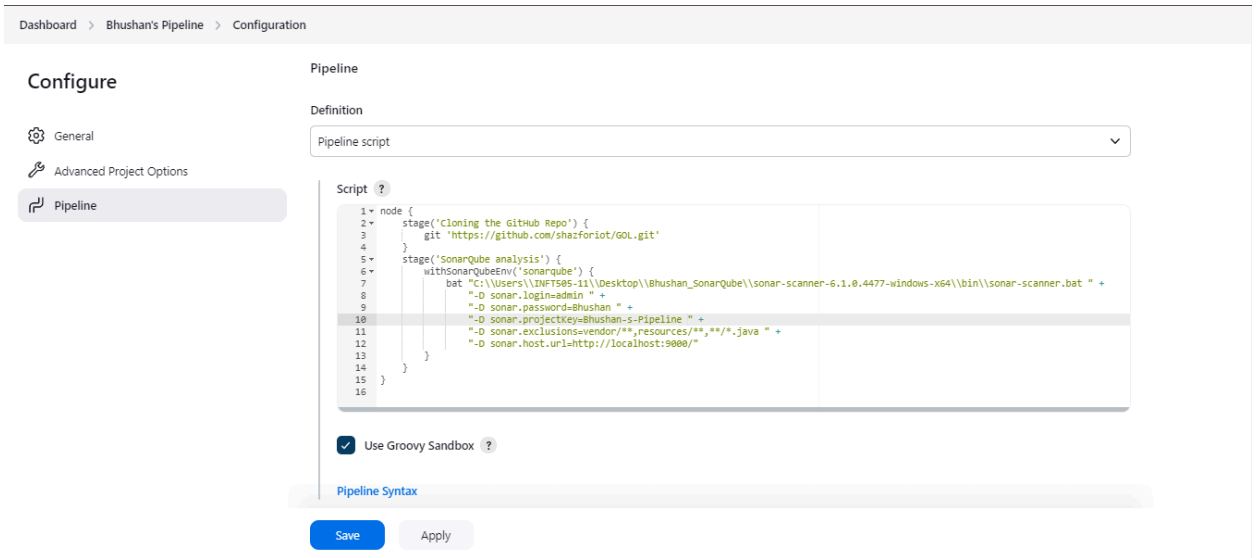
Division: D15C **Roll No: 28**

Step 9: Now click on the new item and select the pipeline project and give name.



Step 10: Under the scripts add the following script.**(Do not forget to replace details with your details.)**

```
node {
        stage('Cloning the GitHub Repo'){
                git 'https://github.com/shazforiot/GOL.git'
        }
        stage('SonarQube analysis') {
                withSonarQubeEnv('sonarqube') {
                        bat  ''<your bin file location of SonarQube CLI>"+
                                "-D sonar.login=<your user name>"+
                                "-D sonar.password=<your password>"+
                                "-D sonar.projectkey=<your projectkey>"+
                                "-D sonar.exclusions=vendor/**,resources/**,**/*.java "+
                                "-D sonar.host.url=http://localhost:9000/"+
                }
        }

}
```

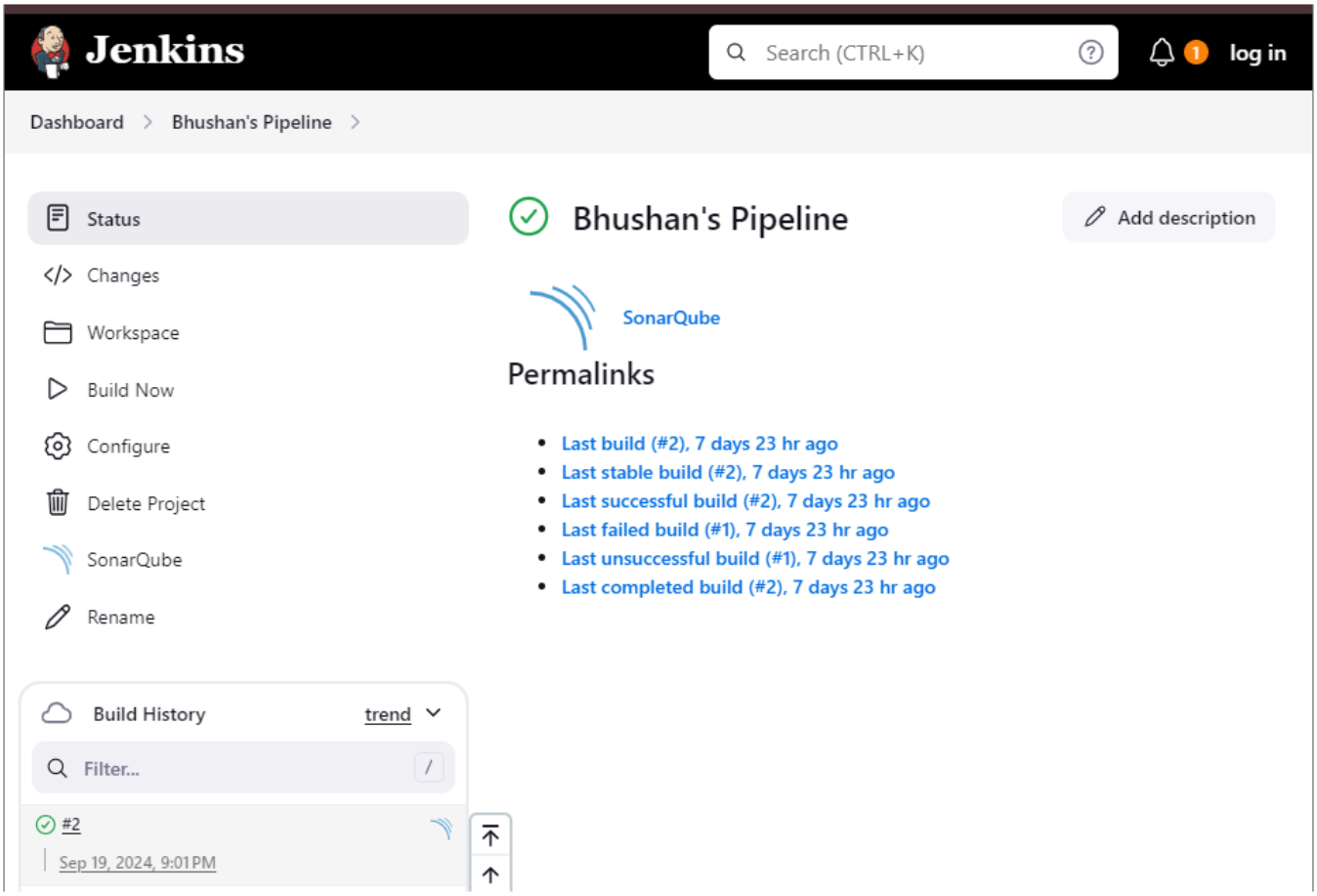


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

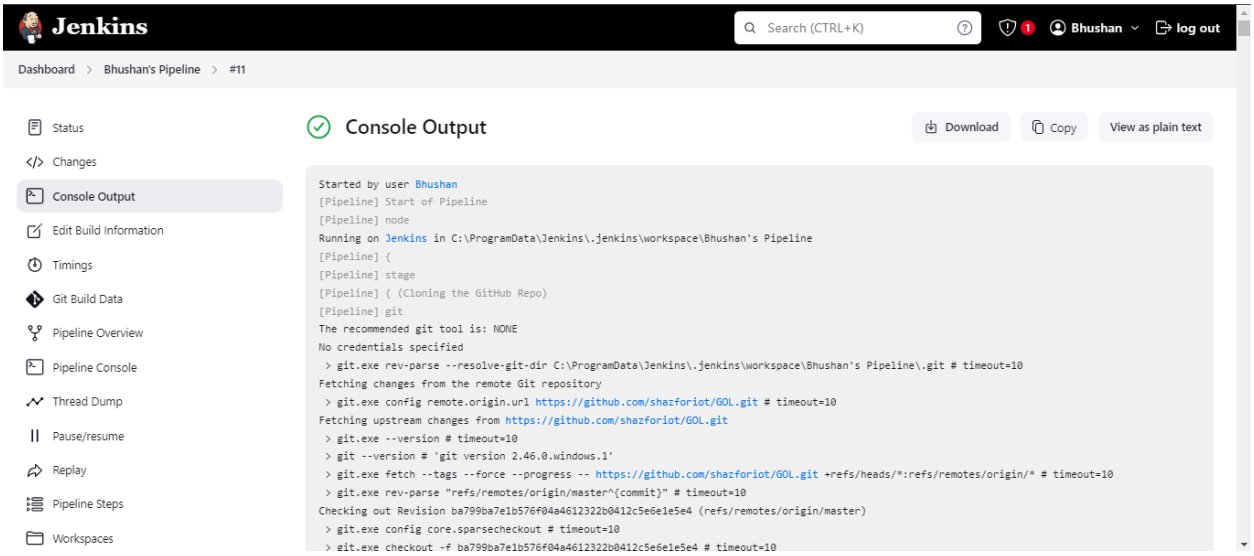
Division: D15C **Roll No: 28**

It is a java sample project which has a lot of repetitions and issues that will be detected by SonarQube.

Step 11: Go back to jenkins. Go to the job you had just built and click on Build Now.



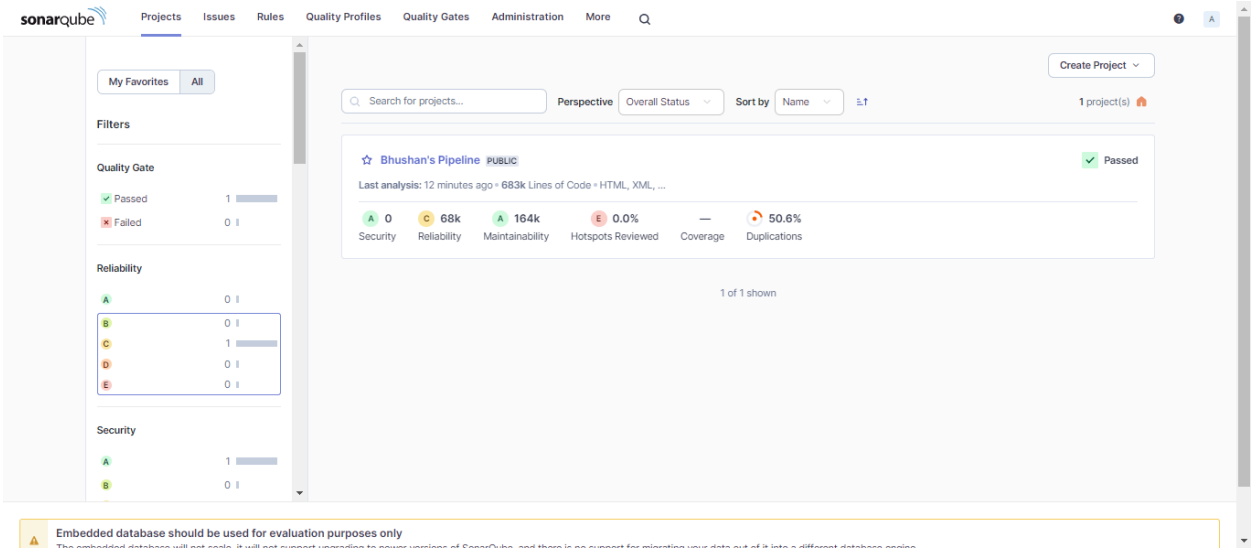
Step 12: Check the console output.

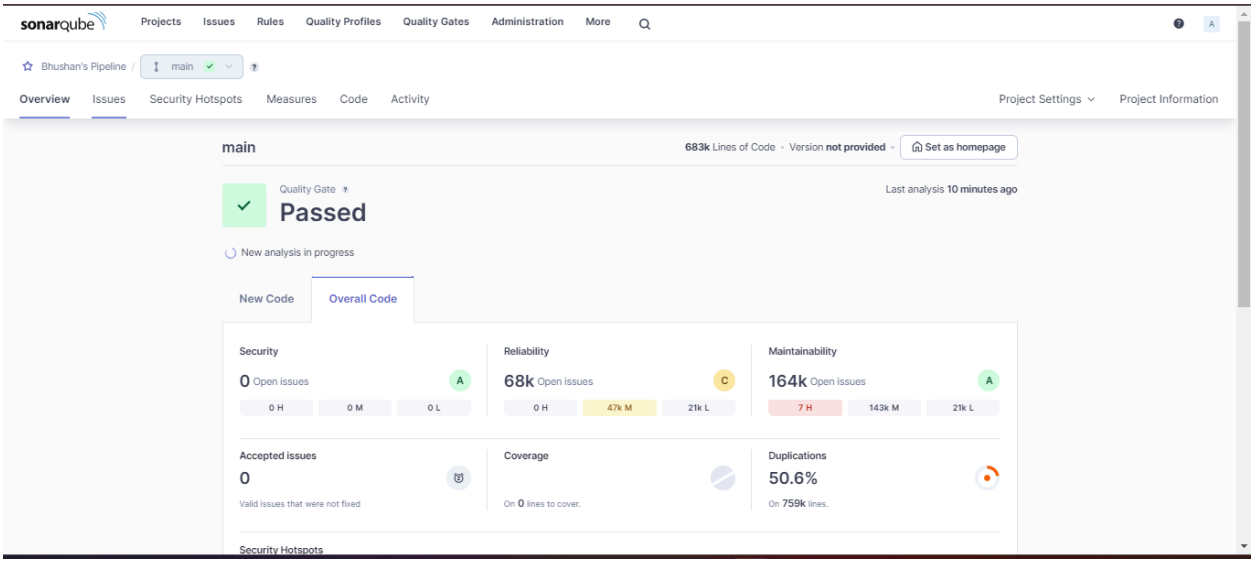


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 13: Once the build is complete, go back to SonarQube and check the project linked.



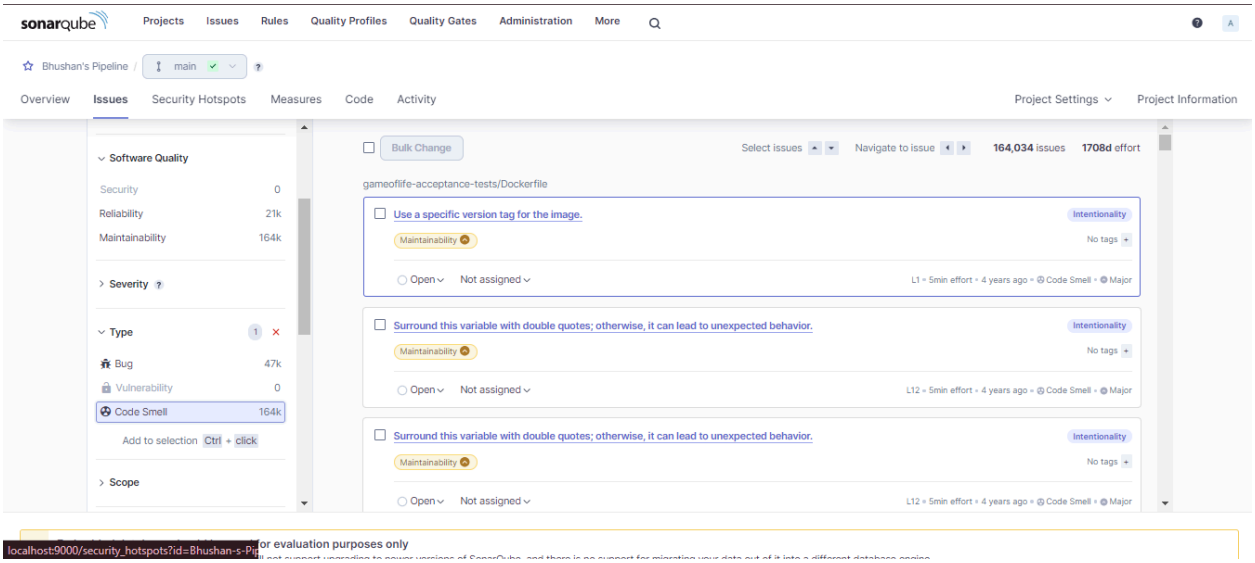


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

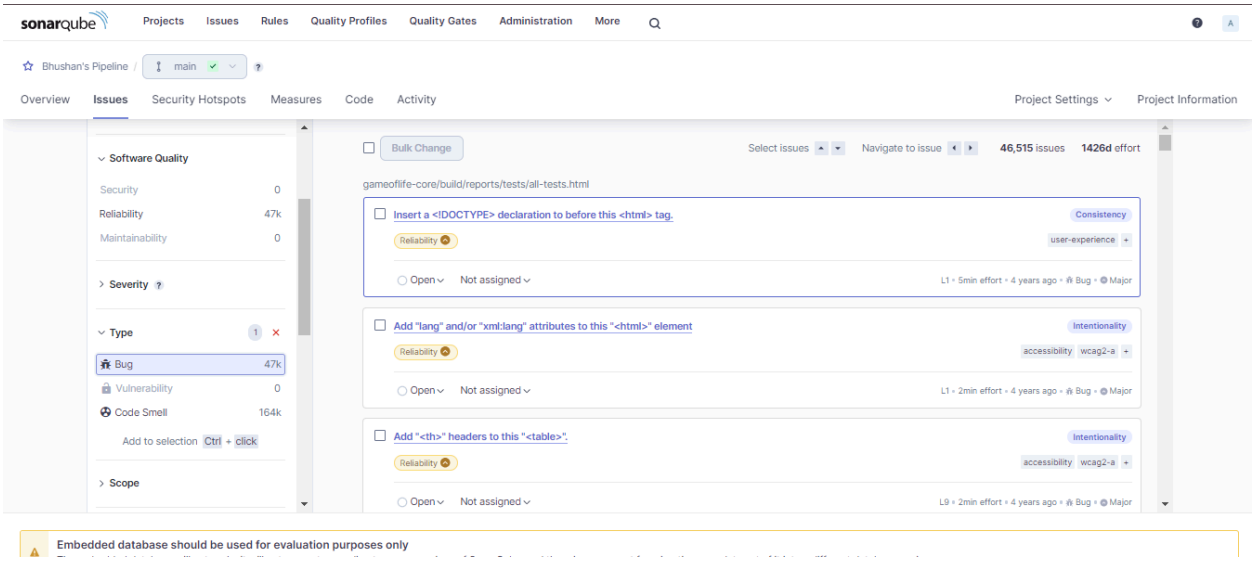
Division: D15C **Roll No: 28**

Click on Issues and see the different Issues.

Codesmell:



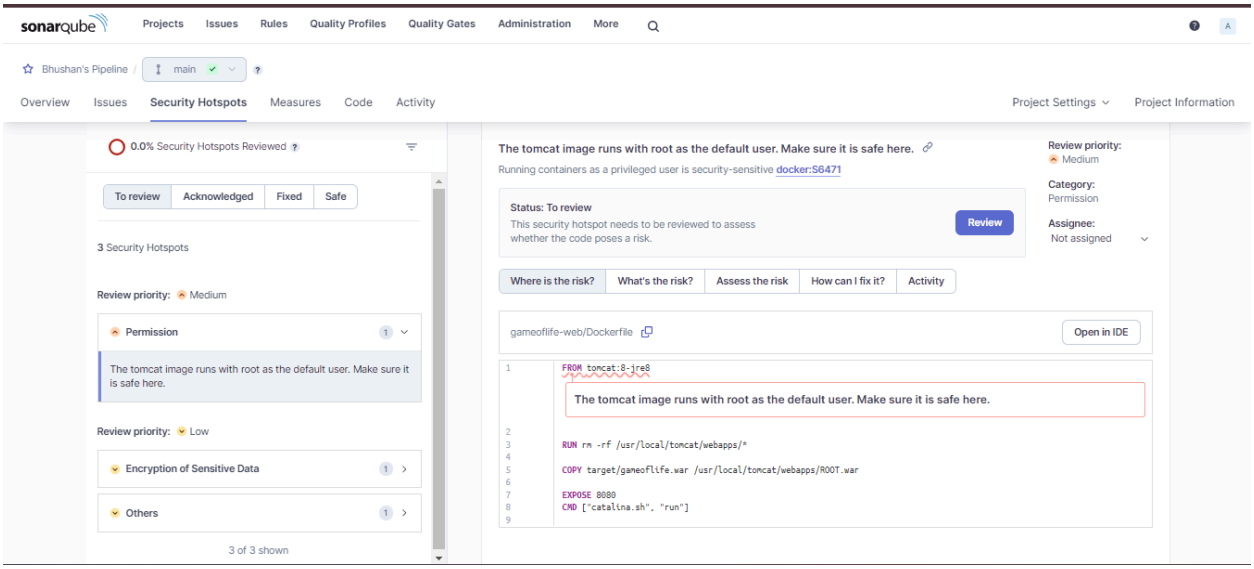
Bugs:



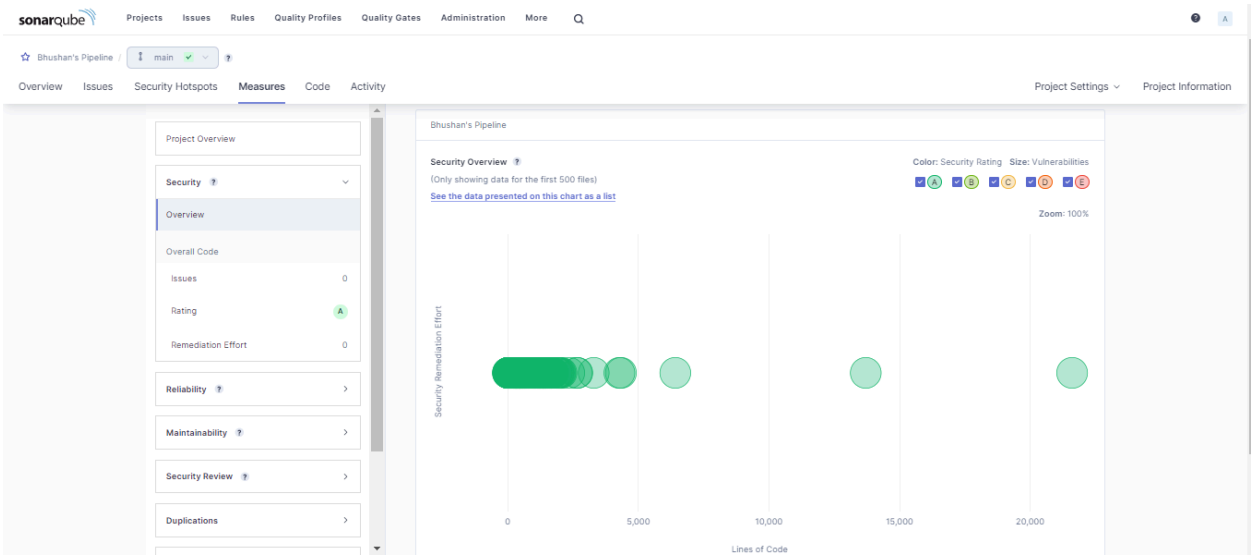
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

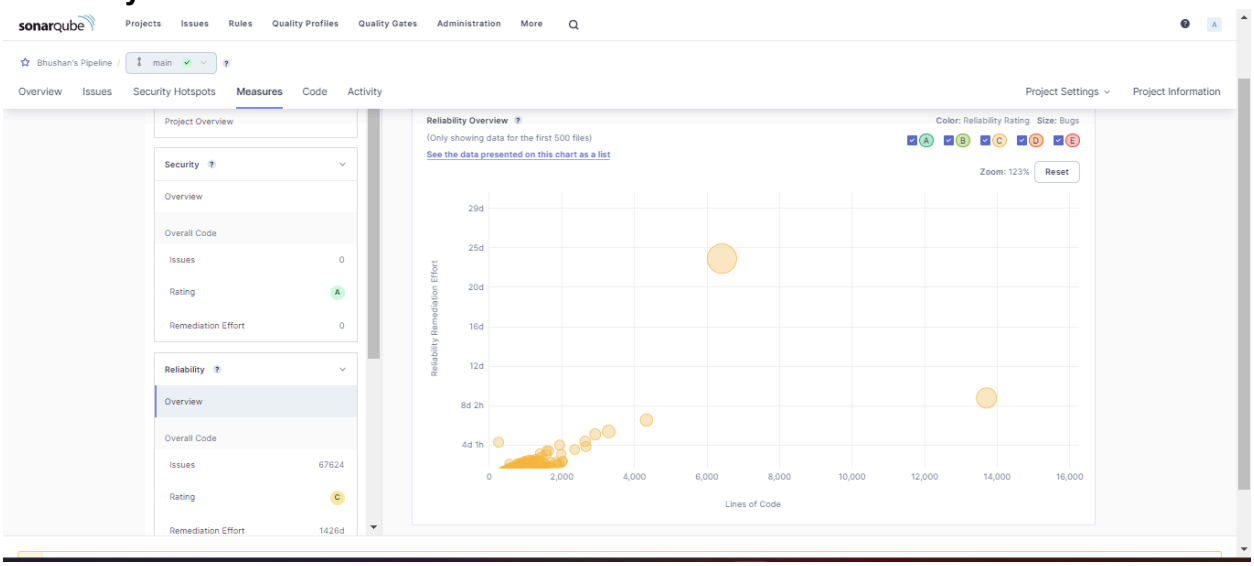
Click on Security hotspots and see the different Security hotspots.



Click on Measures and see the Measures in the form of Graphs.
Security:



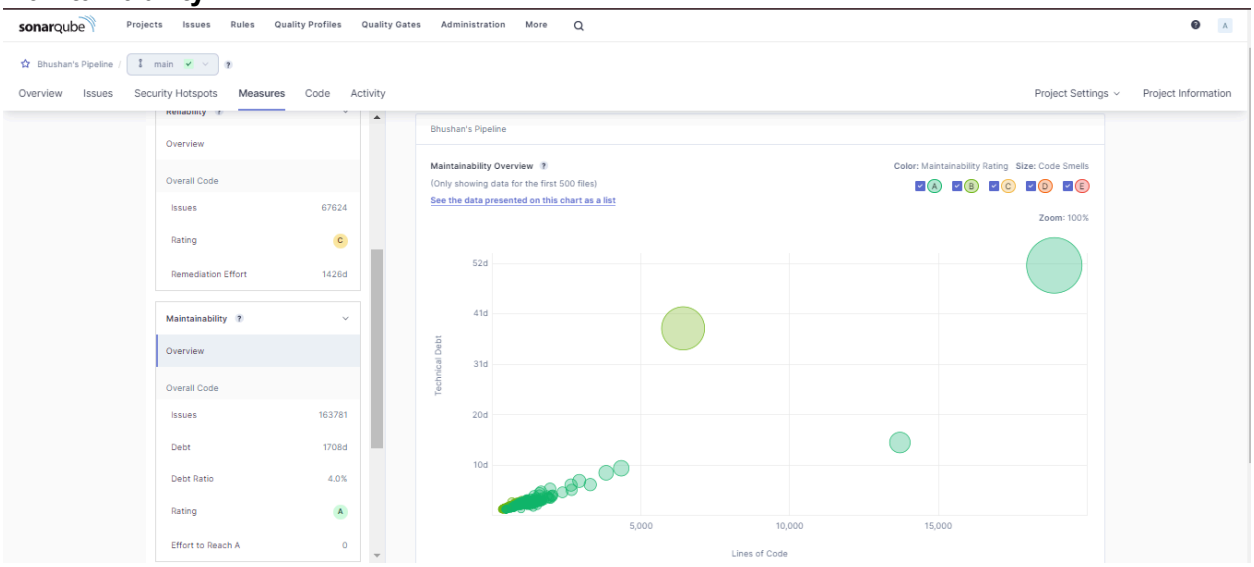
Reliability:



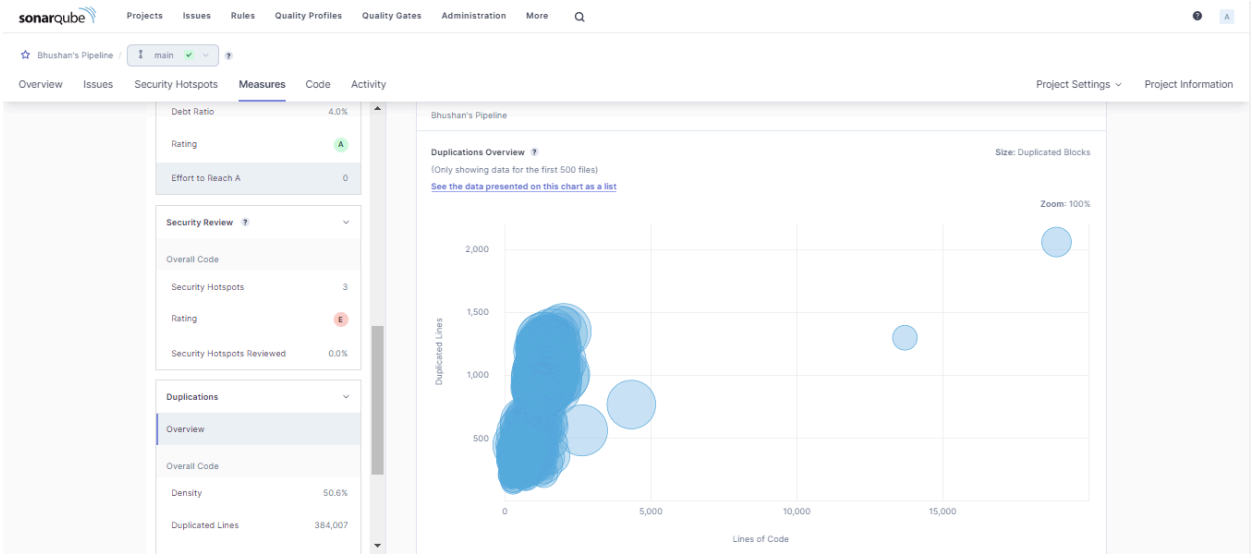
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

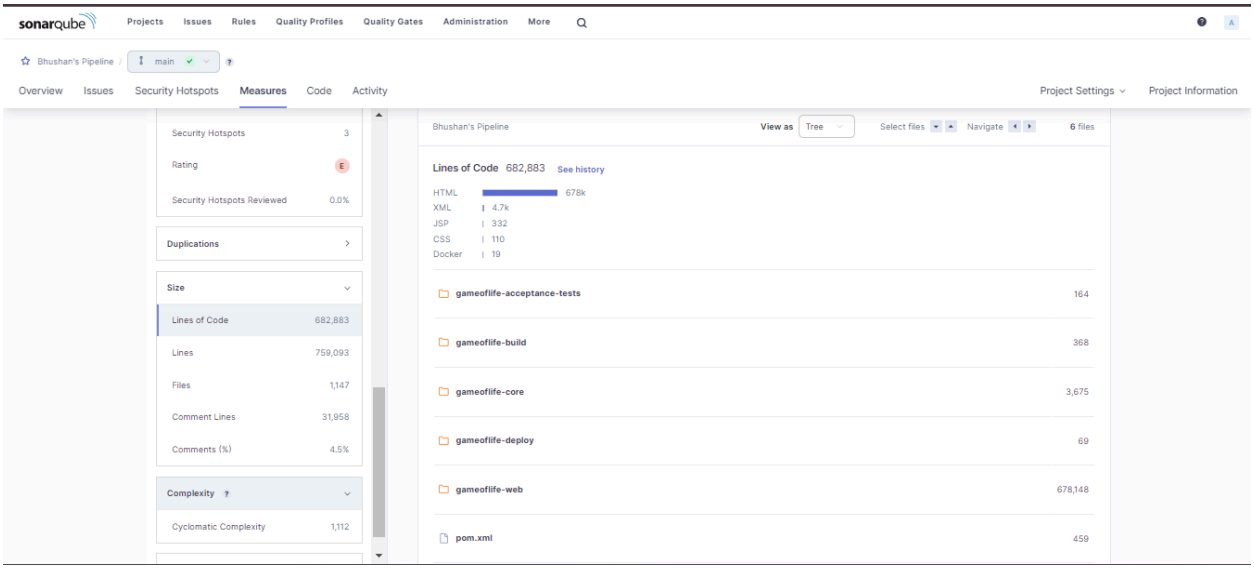
Maintainability:



Duplication:



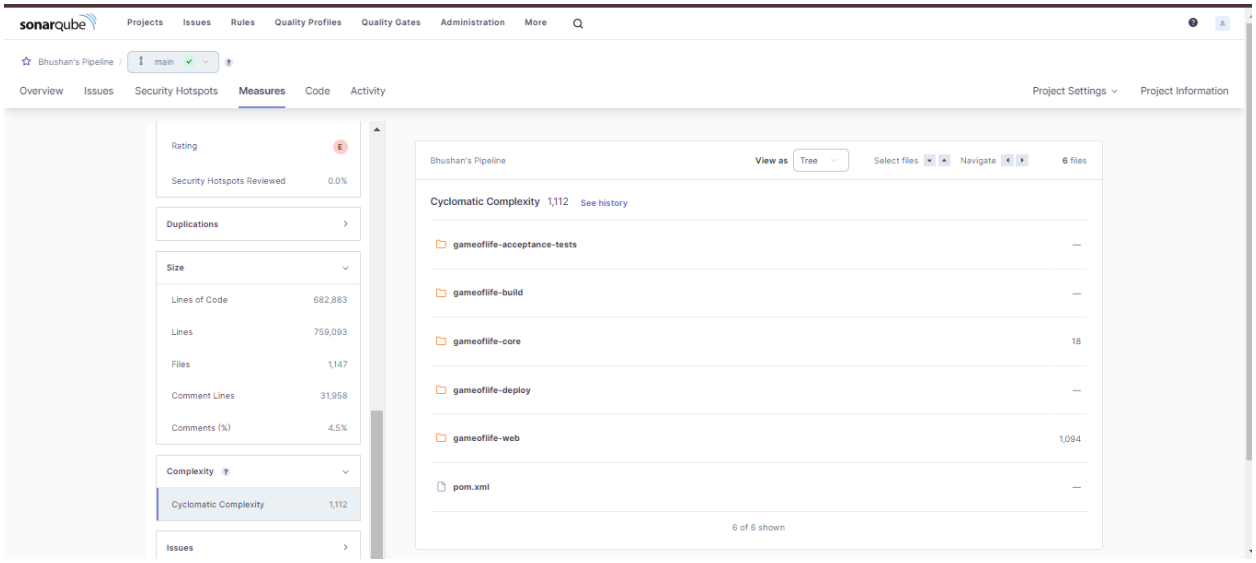
Sizes:



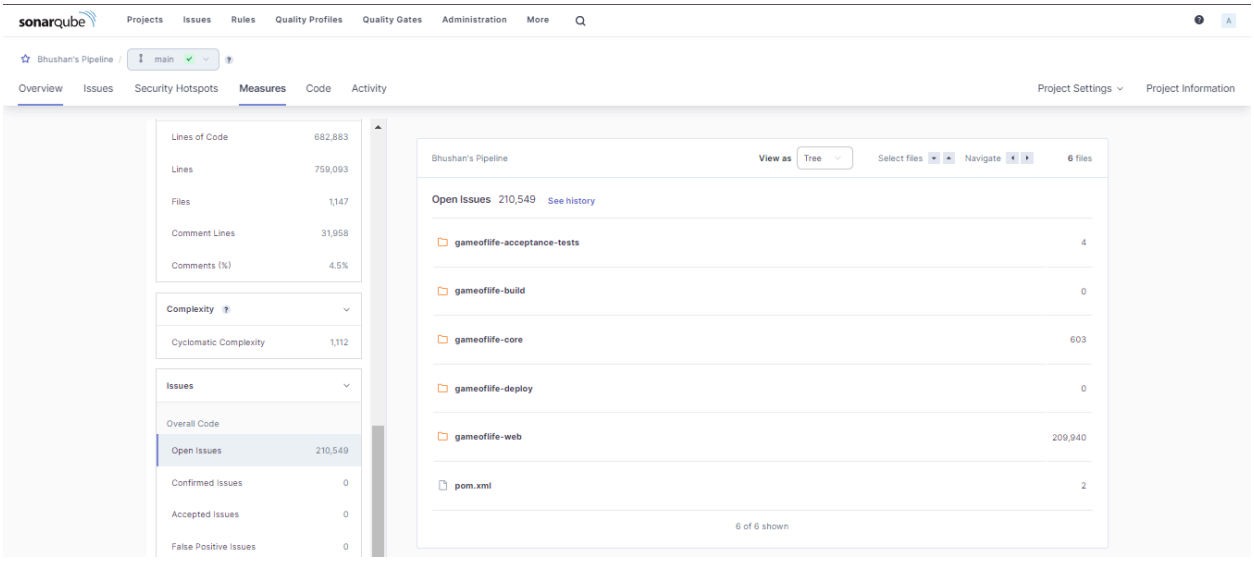
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Complexity:



Issues:



Conclusion: In this experiment, we have learned how to perform static analysis of a code using Jenkins CI/CD Pipeline with SonarQune analysis. A pipeline project is to be created which is given a pipeline script. This script contains all the information needed for the project to run the SonarQube analysis. After the necessary configurations are made on Jenkins, the Jenkins project is built. The code provided in this experiment contains lots of errors, bugs, duplications which can be checked on the SonarQube project linked with this build.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

Theory:

What is Nagios?

Nagios is an open-source software for continuous monitoring of systems, networks, and infrastructures. It runs plugins stored on a server that is connected with a host or another server on your network or the Internet. In case of any failure, Nagios alerts about the issues so that the technical team can perform the recovery process immediately.
Nagios is used for continuous monitoring of systems, applications, service and business processes in a DevOps culture.

Why We Need Nagios tool?

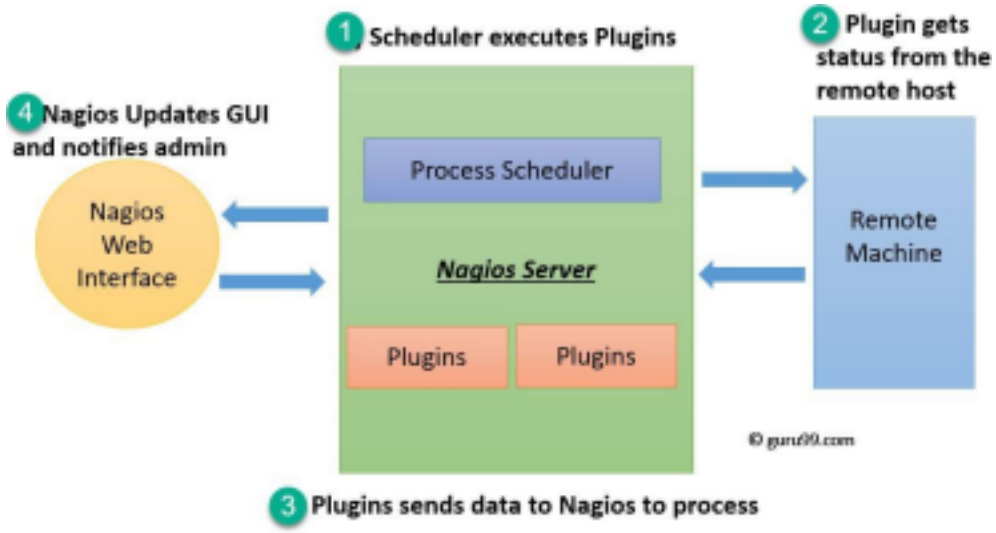
Here are the important reasons to use Nagios monitoring tool:
● Detects all types of network or server issues
● Helps you to find the root cause of the problem which allows you to get the permanent solution to the problem
● Active monitoring of your entire infrastructure and business processes
● Allows you to monitor and troubleshoot server performance issues
● Helps you to plan for infrastructure upgrades before outdated systems create failures
● You can maintain the security and availability of the service
● Automatically fix problems in a panic situation

Features of Nagios
Following are the important features of Nagios monitoring tool:
● Relatively scalable, Manageable, and Secure
● Good log and database system
● Informative and attractive web interfaces
● Automatically send alerts if condition changes
● If the services are running fine, then there is no need to do check that host is an alive
● Helps you to detect network errors or server crashes
● You can troubleshoot the performance issues of the server.
● The issues, if any, can be fixed automatically as they are identified during the monitoring process
● You can monitor the entire business process and IT infrastructure with a single pass
● The product's architecture is easy to write new plugins in the language of your choice
● Nagios allows you to read its configuration from an entire directory which helps you to decide how to define individual files
● Utilizes topology to determine dependencies
● Monitor network services like HTTP, SMTP, HTTP, SNMP, FTP, SSH, POP, etc.
● Helps you to define network host hierarchy using parent hosts

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

● Ability to define event handlers that runs during service or host events for proactive problem resolution
● Support for implementing redundant monitoring hosts
Nagios Architecture
Nagios is a client-server architecture. Usually, on a network, a Nagios server is running on a host, and plugins are running on all the remote hosts which should be monitored.



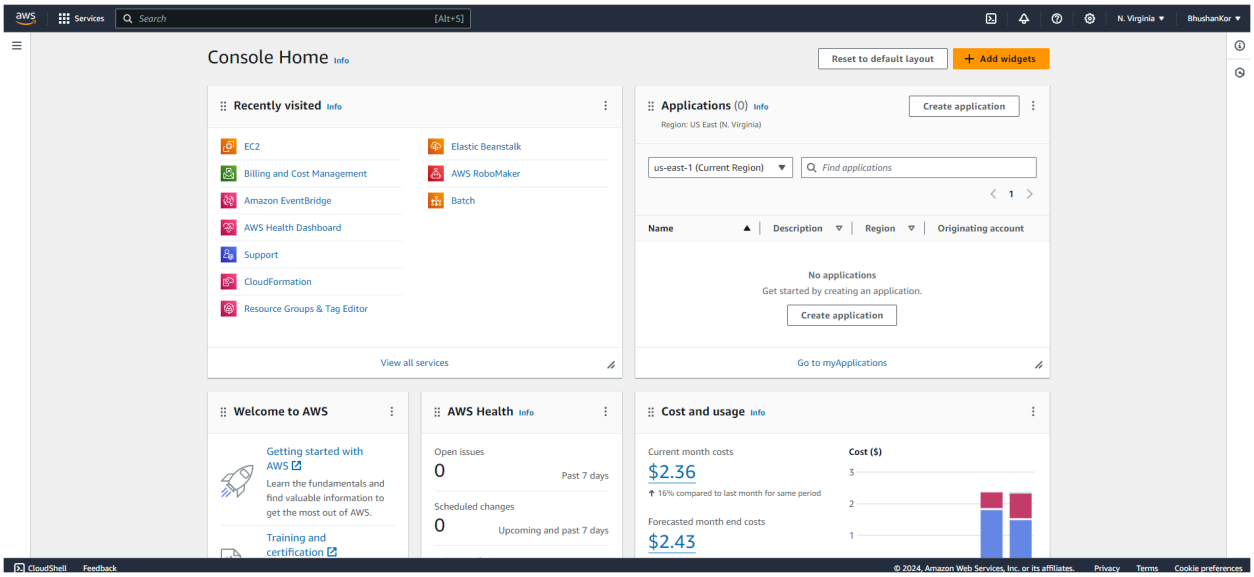
1. The scheduler is a component of the server part of Nagios. It sends a signal to execute the plugins at the remote host.
2. The plugin gets the status from the remote host
3. The plugin sends the data to the process scheduler
4. The process scheduler updates the GUI and notifications are sent to admins.

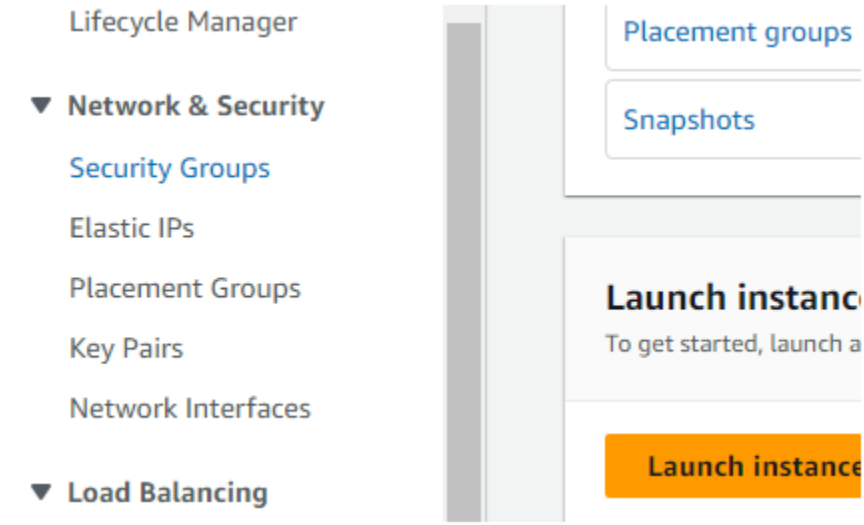
Prerequisites: AWS Personal or Academy Account.

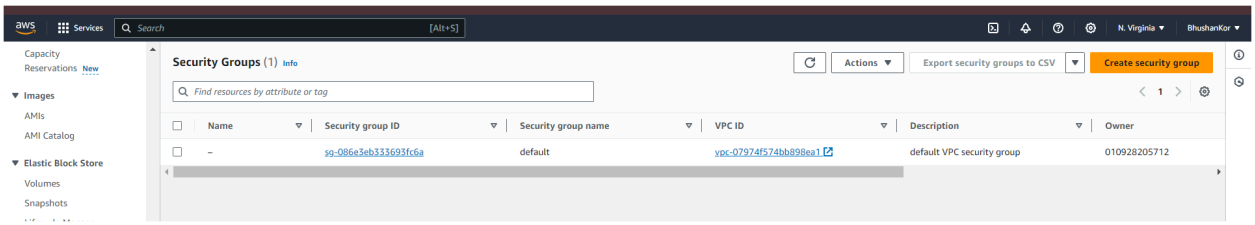
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 1: Login to your AWS account Personal / Academy. Click on EC2 instance then click on Create Security Group. Give the name as Nagios and any description and add the following inbounds rules.

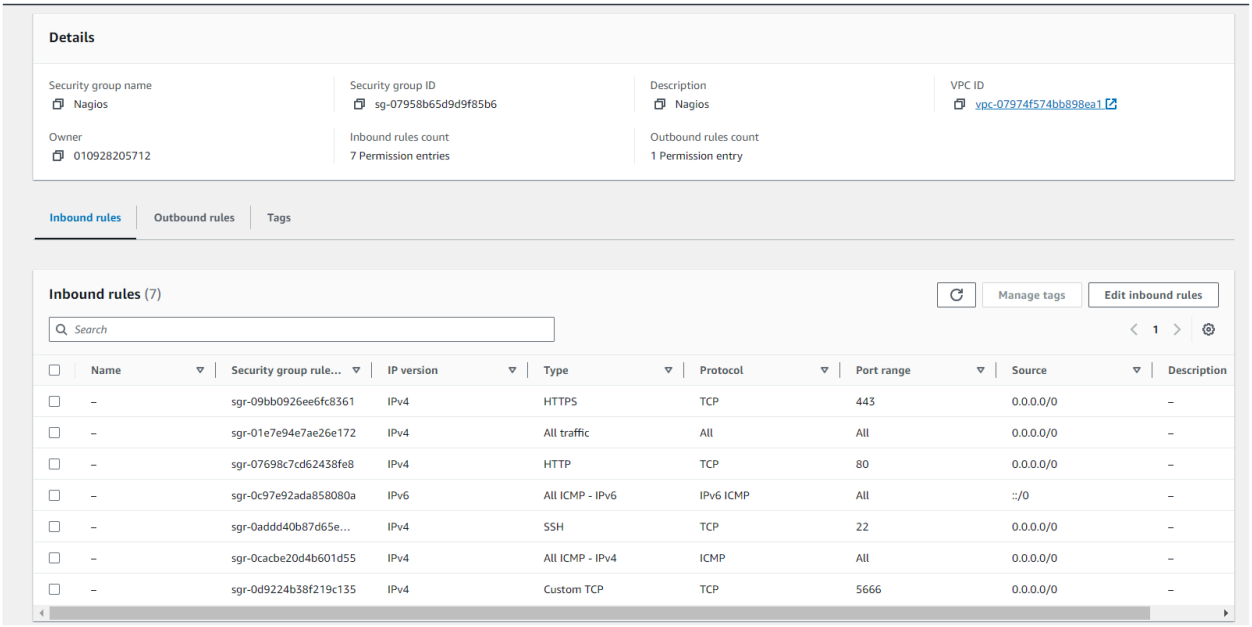




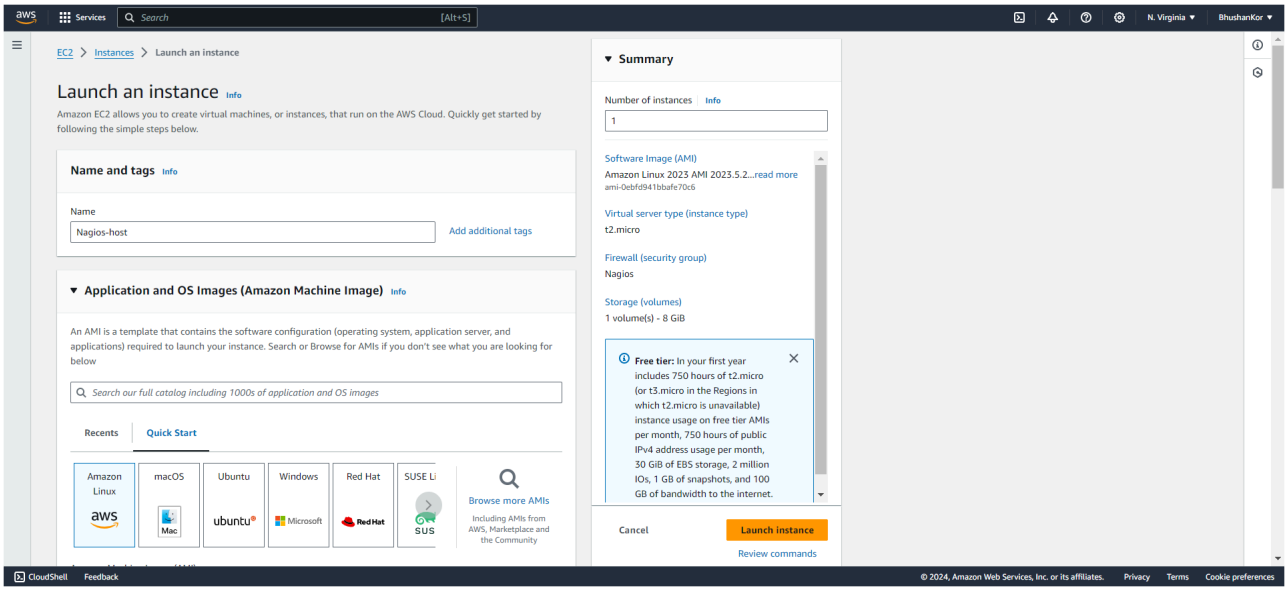


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



Step 2: Now Create a new EC2 instance. Name: Nagios-host ,AMI: Amazon Linux, Instance Type: t2.micro.

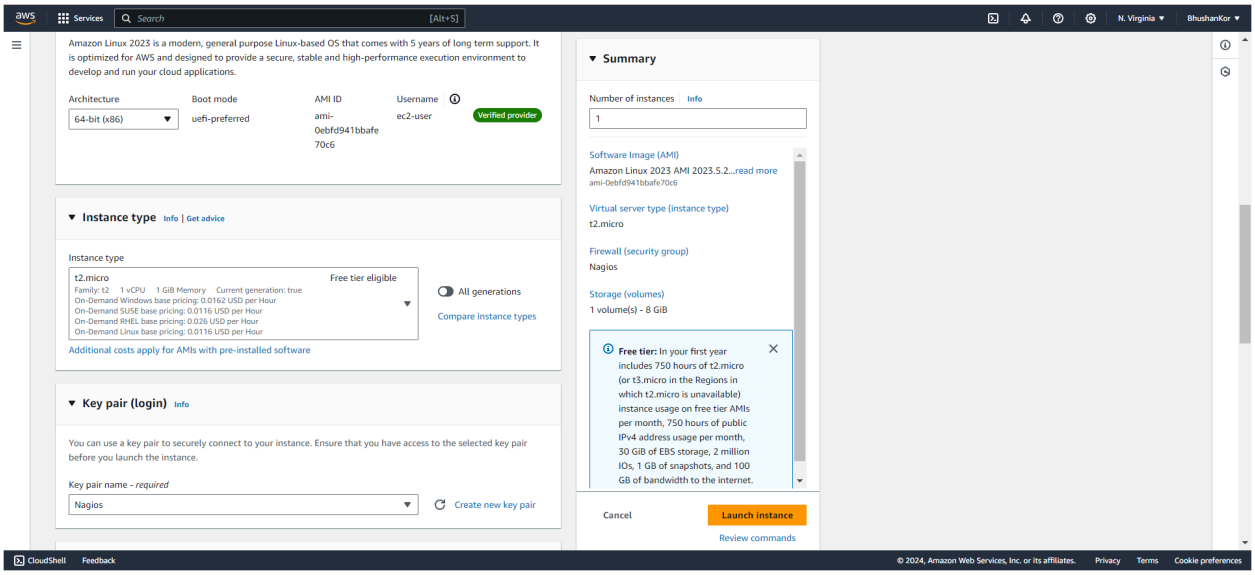


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

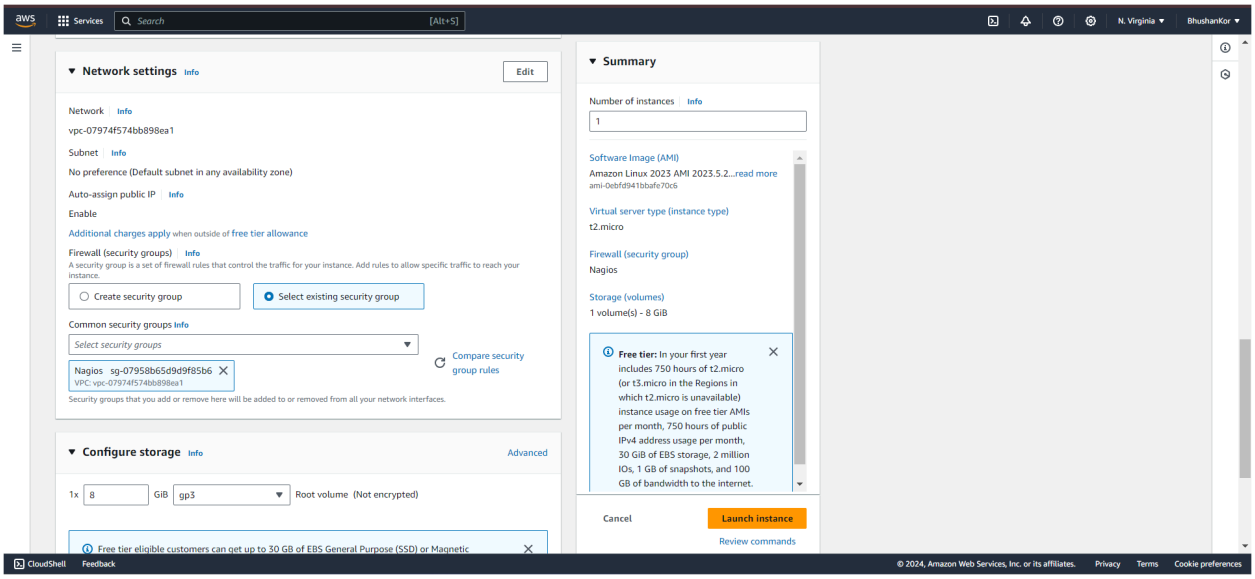
Division: D15C **Roll No: 28**

For Key pair : Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine.

Now select that key in key pair if you already have key with type RSA and extension .pem no need to create new key but you must have that key downloaded.



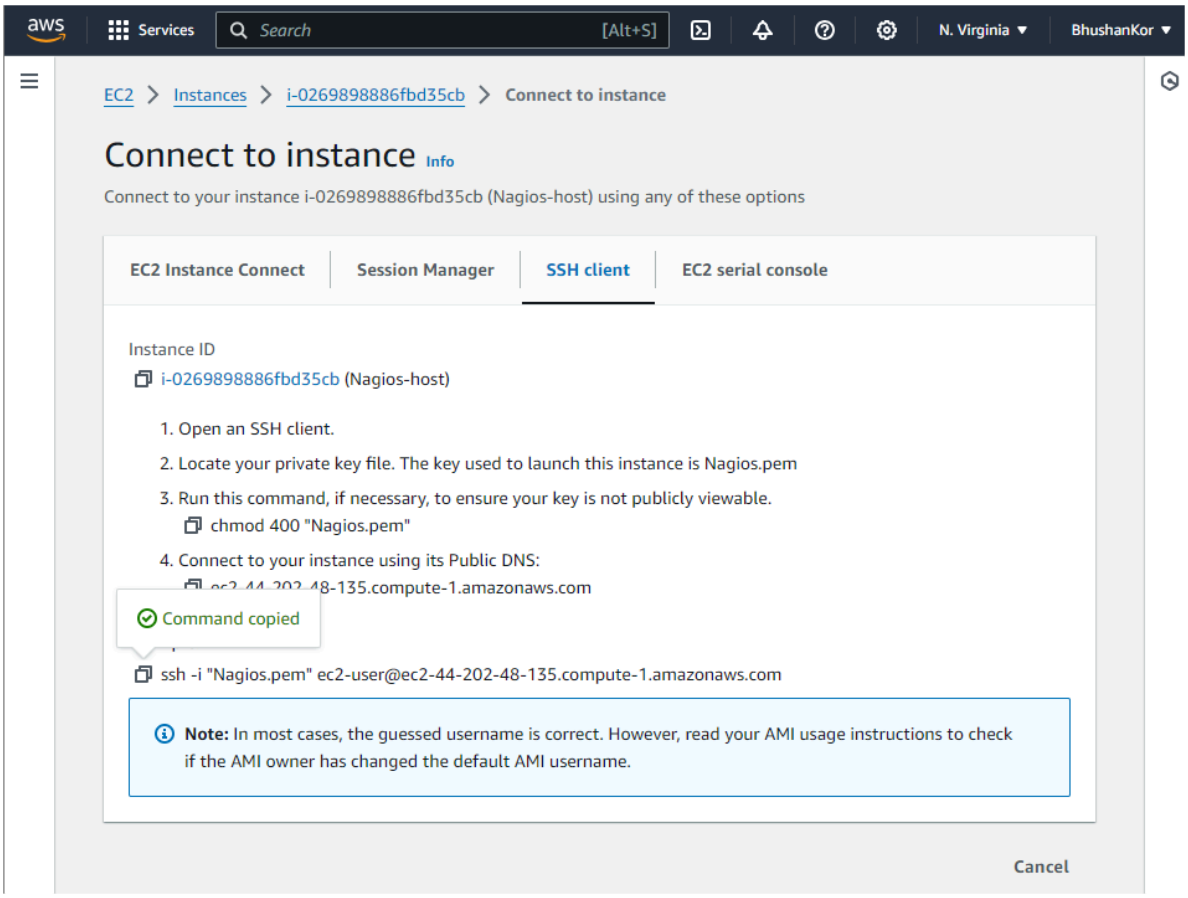
Select the Existing Security Group and select the Security Group we have created in Step 1.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 3: Now After creating the EC2 Instance click on connect and then copy the command which is given as example in the SSH Client section .

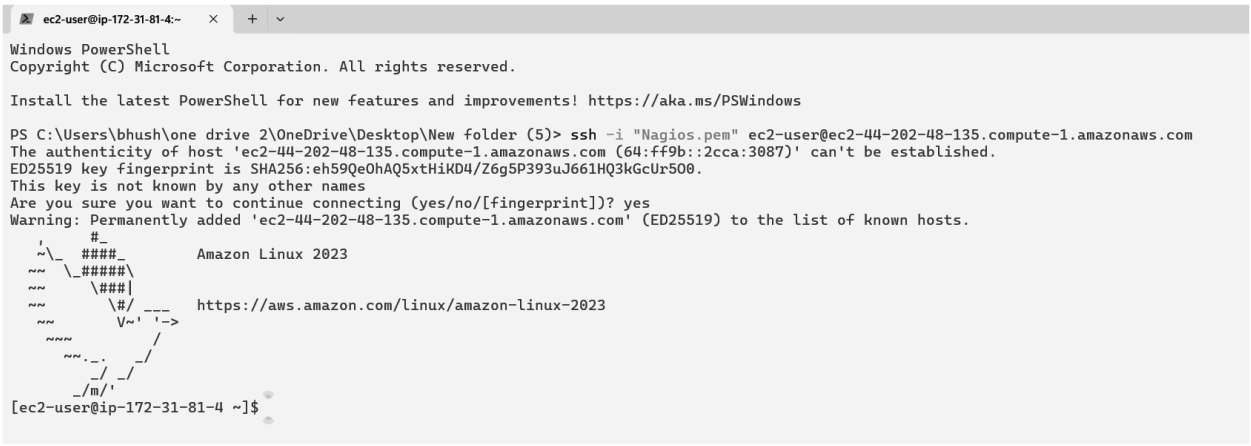


Now open the terminal in the folder where your key(RSA key with .pem) is located.and paste that copied command.

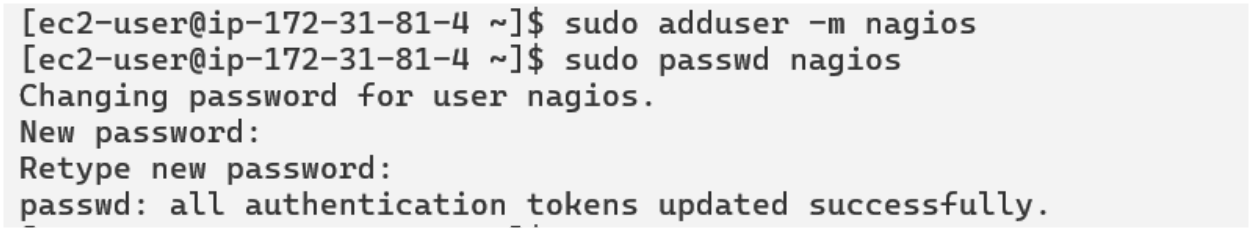
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

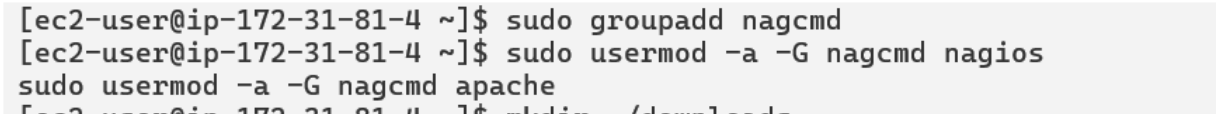
Successfully connected to the instance.



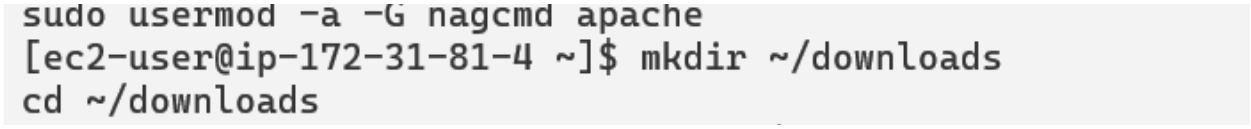
Step 4: Now Run the following command to make a new user.
sudo adduser -m nagios
sudo passwd nagios



Step 5: Now Run the following command to make a new user group.
sudo groupadd nagcmd
sudo usermod -a -G nagcmd nagios
sudo usermod -a -G nagcmd apache



Step 6: Now make a new directory and go to that directory.
mkdir ~/downloads
cd ~/downloads



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 7: Now to download the Nagios 4.5.5 and Nagios-plugins 2.4.11 run the following commands respectively.

wget https://go.nagios.org/l/975333/2024-09-17/6kqcx

```
[ec2-user@ip-172-31-81-4 downloads]$ wget https://go.nagios.org/l/975333/2024-09-17/6kqcx
--2024-09-23 15:43:53--  https://go.nagios.org/l/975333/2024-09-17/6kqcx
Resolving go.nagios.org (go.nagios.org)... 34.237.219.119, 52.54.96.194, 18.208.125.13, ...
Connecting to go.nagios.org (go.nagios.org)|34.237.219.119|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.5
+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-09-23 15:43:53--  http://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_camp
aign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
Resolving assets.nagios.com (assets.nagios.com)... 45.79.49.120, 2600:3c00::f03c:92ff:fef7:45ce
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_campaign=Core+4.5.
5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8 [following]
--2024-09-23 15:43:53--  https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.5.5.tar.gz?utm_source=Nagios.org&utm_content=Download+Form&utm_cam
paign=Core+4.5.5+Download+&pi_content=1e9662c93afb2ed6bd2e3f3cc38771a7f01125e969f2a75b0e2254439d4a81d8
Connecting to assets.nagios.com (assets.nagios.com)|45.79.49.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2065473 (2.0M) [application/x-gzip]
Saving to: ‘6kqcx’

6kqcx             100%[================>]   1.97M  7.19MB/s    in 0.3s

2024-09-23 15:43:54 (7.19 MB/s) - ‘6kqcx’ saved [2065473/2065473]
```

wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz

```
[ec2-user@ip-172-31-81-4 downloads]$ wget https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
--2024-09-23 15:44:01--  https://nagios-plugins.org/download/nagios-plugins-2.4.11.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Connecting to nagios-plugins.org (nagios-plugins.org)|45.56.123.251|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2753049 (2.6M) [application/x-gzip]
Saving to: ‘nagios-plugins-2.4.11.tar.gz’

nagios-plugins-2.4 100%[=================>]   2.62M  6.70MB/s    in 0.4s

2024-09-23 15:44:01 (6.70 MB/s) - ‘nagios-plugins-2.4.11.tar.gz’ saved [2753049/2753049]
```

Step 8: Now to extract the files from the downloaded Nagios 4.5.5 run the following command.

tar zxvf 6kqcx (Replace 6kqcx with your saved file name of Nagios 4.5.5 refer above screenshot(1st))

```
[ec2-user@ip-172-31-81-4 downloads]$ tar zxvf 6kqcx
nagios-4.5.5/
nagios-4.5.5/.github/
nagios-4.5.5/.github/workflows/
nagios-4.5.5/.github/workflows/test.yml
nagios-4.5.5/.gitignore
nagios-4.5.5/CONTRIBUTING.md
nagios-4.5.5/Changelog
nagios-4.5.5/INSTALLING
nagios-4.5.5/LEGAL
nagios-4.5.5/LICENSE
nagios-4.5.5/Makefile.in
nagios-4.5.5/README.md
nagios-4.5.5/THANKS
nagios-4.5.5/UPGRADING
nagios-4.5.5/aclocal.m4
nagios-4.5.5/autoconf-macros/
nagios-4.5.5/autoconf-macros/.gitignore
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 9: Now change the directory to nagios-4.5.5 (Or which version you have downloaded)

```
[ec2-user@ip-172-31-81-4 downloads]$ cd nagios-4.5.5
```

Step 10: Now run the following command to configure.
./configure --with-command-group=nagcmd

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether the compiler supports GNU C... yes
checking whether gcc accepts -g... yes
checking for gcc option to enable C11 features... none needed
checking whether make sets $(MAKE)... yes
checking whether ln -s works... yes
checking for strip... /usr/bin/strip
checking for sys/wait.h that is POSIX.1 compatible... yes
checking for stdio.h... yes
checking for stdlib.h... yes
checking for string.h... yes
```

At the end we have found the error of cannot find ssl header .

```
checking for pkg-config... pkg-config
checking for SSL headers... configure: error: Cannot find ssl headers
```

So run following command to install ssl.
sudo yum install openssl-devel

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo yum install openssl-devel
Last metadata expiration check: 0:10:45 ago on Mon Sep 23 15:36:08 2024.
Dependencies resolved.
================================================================================
 Package          Arch      Version                    Repository      Size
================================================================================
Installing:
 openssl-devel    x86_64    1:3.0.8-1.amzn2023.0.14    amazonlinux    3.0 M

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

Total download size: 3.0 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
openssl-devel-3.0.8-1.amzn2023.0.14.x86_64.  35 MB/s | 3.0 MB     00:00
--------------------------------------------------------------------------------
Total                                        24 MB/s | 3.0 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64           1/1
  Running scriptlet: openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64           1/1
  Verifying        : openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64           1/1

Installed:
  openssl-devel-1:3.0.8-1.amzn2023.0.14.x86_64

Complete!
```

Now rerun the command **./configure --with-command-group=nagcmd**

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ ./configure --with-command-group=nagcmd
checking for a BSD-compatible install... /usr/bin/install -c
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
```

```
     - This installs the Exfoliation theme for the Nagios
       web interface

  make install-classicui
     - This installs the classic theme for the Nagios
       web interface


*** Support Notes *******************************************

If you have questions about configuring or running Nagios,
please make sure that you:

     - Look at the sample config files
     - Read the documentation on the Nagios Library at:
           https://library.nagios.com

before you post a question to one of the mailing lists.
Also make sure to include pertinent information that could
help others help you.  This might include:

     - What version of Nagios you are using
     - What version of the plugins you are using
     - Relevant snippets from your config files
     - Relevant error messages from the Nagios log file

For more information on obtaining support for Nagios, visit:

       https://support.nagios.com

************************************************************

Enjoy.
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 11: Now run the following commands to steup the Nagios.
sudo make install

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo make install
cd ./base && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/base'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagios /usr/local/nagios/bin
/usr/bin/install -c -s -m 774 -o nagios -g nagios nagiostats /usr/local/nagios/bin
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/base'
cd ./cgi && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make install-basic
make[2]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/sbin
for file in *.cgi; do \
        /usr/bin/install -c -s -m 775 -o nagios -g nagios $file /usr/local/nagios/sbin; \
done
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5/cgi'
cd ./html && make install
make[1]: Entering directory '/home/ec2-user/downloads/nagios-4.5.5/html'
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/media
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/stylesheets
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/contexthelp
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/docs/images
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/js
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/images
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/images/logos
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/includes
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/share/ssi
/usr/bin/install -c -m 664 -o nagios -g nagios ./robots.txt /usr/local/nagios/share
/usr/bin/install -c -m 664 -o nagios -g nagios ./jsonquery.html /usr/local/nagios/share
rm -f /usr/local/nagios/share/index.html
rm -f /usr/local/nagios/share/main.html
rm -f /usr/local/nagios/share/side.html
rm -f /usr/local/nagios/share/map.html
```

```
/usr/bin/install -c -m 775 -o nagios -g nagcmd -d /usr/local/nagios/var/spool/checkresults
chmod g+s /usr/local/nagios/var/spool/checkresults

*** Main program, CGIs and HTML files installed ***

You can continue with installing Nagios as follows (type 'make'
without any arguments for a list of all possible options):

  make install-init
     - This installs the init script in /lib/systemd/system

  make install-commandmode
     - This installs and configures permissions on the
       directory for holding the external command file

  make install-config
     - This installs sample config files in /usr/local/nagios/etc

make[1]: Leaving directory '/home/ec2-user/downloads/nagios-4.5.5'
```

sudo make install-init

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo make install-init
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo make install-config
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

sudo make install-config

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo make install-config
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/objects
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.cfg /usr/local/nagios/etc/nagios.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cfg /usr/local/nagios/etc/cgi.cfg
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource.cfg /usr/local/nagios/etc/resource.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/templates.cfg /usr/local/nagios/etc/objects/templates.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/commands.cfg /usr/local/nagios/etc/objects/commands.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/contacts.cfg /usr/local/nagios/etc/objects/contacts.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/timeperiods.cfg /usr/local/nagios/etc/objects/timeperiods.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/localhost.cfg /usr/local/nagios/etc/objects/localhost.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/windows.cfg /usr/local/nagios/etc/objects/windows.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/printer.cfg /usr/local/nagios/etc/objects/printer.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/switch.cfg /usr/local/nagios/etc/objects/switch.cfg

*** Config files installed ***
```

sudo make install-webconf

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo make install-webconf
/usr/bin/install -c -m 644 sample-config/httpd.conf /etc/httpd/conf.d/nagios.conf
if [ 0 -eq 1 ]; then \
        ln -s /etc/httpd/conf.d/nagios.conf /etc/apache2/sites-enabled/nagios.conf; \
fi

*** Nagios/Apache conf file installed ***
```

sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin **(To set the password)**

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

Now to restart the httpd service run the following command.

sudo service httpd restart

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ sudo service httpd restart
Redirecting to /bin/systemctl restart httpd.service
```

Step 12: Now to extract the files from the downloaded Nagios plugin 2.4.11 run the following command first change the directory.

cd ~/downloads

tar zxvf nagios-plugins-2.4.11.tar.gz (According to your version)

```
[ec2-user@ip-172-31-81-4 nagios-4.5.5]$ cd ~/downloads
[ec2-user@ip-172-31-81-4 downloads]$ tar zxvf nagios-plugins-2.4.11.tar.gz
nagios-plugins-2.4.11/
nagios-plugins-2.4.11/build-aux/
nagios-plugins-2.4.11/build-aux/compile
nagios-plugins-2.4.11/build-aux/config.guess
nagios-plugins-2.4.11/build-aux/config.rpath
nagios-plugins-2.4.11/build-aux/config.sub
nagios-plugins-2.4.11/build-aux/install-sh
nagios-plugins-2.4.11/build-aux/ltmain.sh
nagios-plugins-2.4.11/build-aux/missing
nagios-plugins-2.4.11/build-aux/mkinstalldirs
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 13: Now change the directory to nagios-plugins-2.4.11 and run the config command to configure.
cd nagios-plugins-2.4.11

./configure --with-nagios-user=nagios --with-nagios-group=nagios

```
nagios-plugins-2.4.11/po/ChangeLog
nagios-plugins-2.4.11/po/LINGUAS
nagios-plugins-2.4.11/release
[ec2-user@ip-172-31-81-4 downloads]$ cd nagios-plugins-2.4.11
[ec2-user@ip-172-31-81-4 nagios-plugins-2.4.11]$ ./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking whether to enable maintainer-specific portions of Makefiles... yes
checking build system type... x86_64-pc-linux-gnu
```

Step 14: Run the following commands to check nagios and start it.
sudo chkconfig --add nagios

```
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
make[1]: Leaving directory '/home/ec2-user/downloads/nagios-plugins-2.4.11'
[ec2-user@ip-172-31-81-4 nagios-plugins-2.4.11]$ sudo chkconfig --add nagios
sudo chkconfig nagios on
error reading information on service nagios: No such file or directory
Note: Forwarding request to 'systemctl enable nagios.service'.
Created symlink /etc/systemd/system/multi-user.target.wants/nagios.service → /usr/lib/systemd/system/nagios.service.
```

sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

```
[ec2-user@ip-172-31-81-4 nagios-plugins-2.4.11]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
        Checked 8 services.
        Checked 1 hosts.
        Checked 1 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 1 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

cd
sudo service nagios start

```
[ec2-user@ip-172-31-81-4 nagios-plugins-2.4.11]$ cd
[ec2-user@ip-172-31-81-4 ~]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
```

sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

```
[ec2-user@ip-172-31-81-4 ~]$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
        Checked 8 services.
        Checked 1 hosts.
        Checked 1 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 1 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
[ec2-user@ip-172-31-81-4 ~]$ sudo service nagios start
Redirecting to /bin/systemctl start nagios.service
```

sudo systemctl status nagios

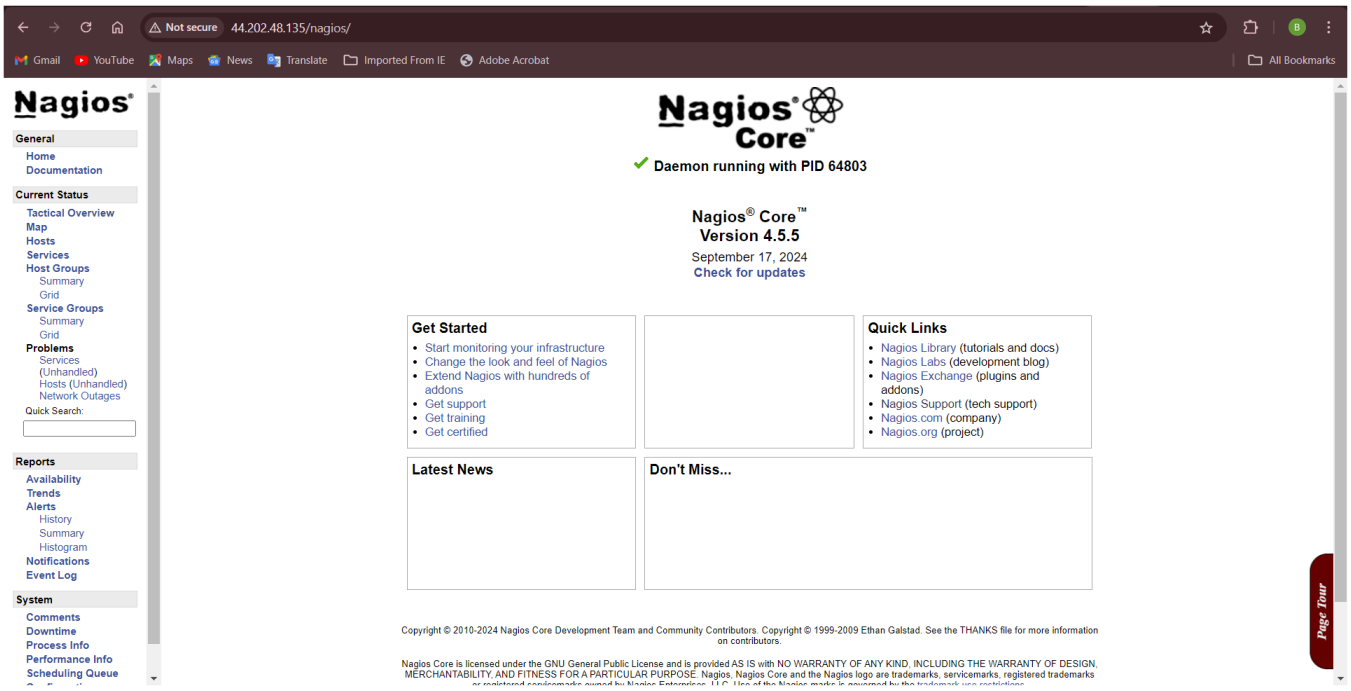
```
[ec2-user@ip-172-31-81-4 ~]$ sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; prese>
     Active: active (running) since Mon 2024-09-23 16:00:54 UTC; 1min 18s a>
       Docs: https://www.nagios.org/documentation
    Process: 64801 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/>
    Process: 64802 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nag>
   Main PID: 64803 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.9M
        CPU: 95ms
     CGroup: /system.slice/nagios.service
             ├─64803 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/>
             ├─64804 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>
             ├─64805 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>
             ├─64806 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>
             ├─64807 /usr/local/nagios/bin/nagios --worker /usr/local/nagio>
             └─64808 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/>

Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: Socket '/usr>
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: core query h>
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: echo service>
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: help for the>
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Successfu>
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry >
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry >
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry >
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry >
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: Successfully lau>
lines 1-28/28 (END)
● nagios.service - Nagios Core 4.5.5
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

```
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Mon 2024-09-23 16:00:54 UTC; 1min 18s ago
       Docs: https://www.nagios.org/documentation
    Process: 64801 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 64802 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 64803 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 5.9M
        CPU: 95ms
     CGroup: /system.slice/nagios.service
             ├─64803 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─64804 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─64805 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─64806 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─64807 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─64808 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: Socket '/usr/local/nagios/var/rw/nagios.qh' successfully initialized
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: core query handler registered
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: echo service query handler registered
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: qh: help for the query handler registered
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Successfully registered manager as @wproc with query handler
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry request: name=Core Worker 64807;pid=64807
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry request: name=Core Worker 64806;pid=64806
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry request: name=Core Worker 64805;pid=64805
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: wproc: Registry request: name=Core Worker 64804;pid=64804
Sep 23 16:00:54 ip-172-31-81-4.ec2.internal nagios[64803]: Successfully launched command file worker with pid 64808
~
~
~
~
~
~
~
~
~
~
~
~
```

Step 15: We can see we have successfully launched the Nagios now . Open **http://<instance public ip >/nagios/** here it is http://44.202.48.135/nagios we can see the running web page of nagios.



Conclusion: In this experiment, we have setup the Nagios core with plugins on Amazon Linux. Which will help us to understand Continuous monitoring and Installation. It is important to note that whatever set of rules we have added in step 1 are very important for this experiment.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To perform Port, Service monitoring, and Windows/Linux server monitoring using Nagios.

Theory:

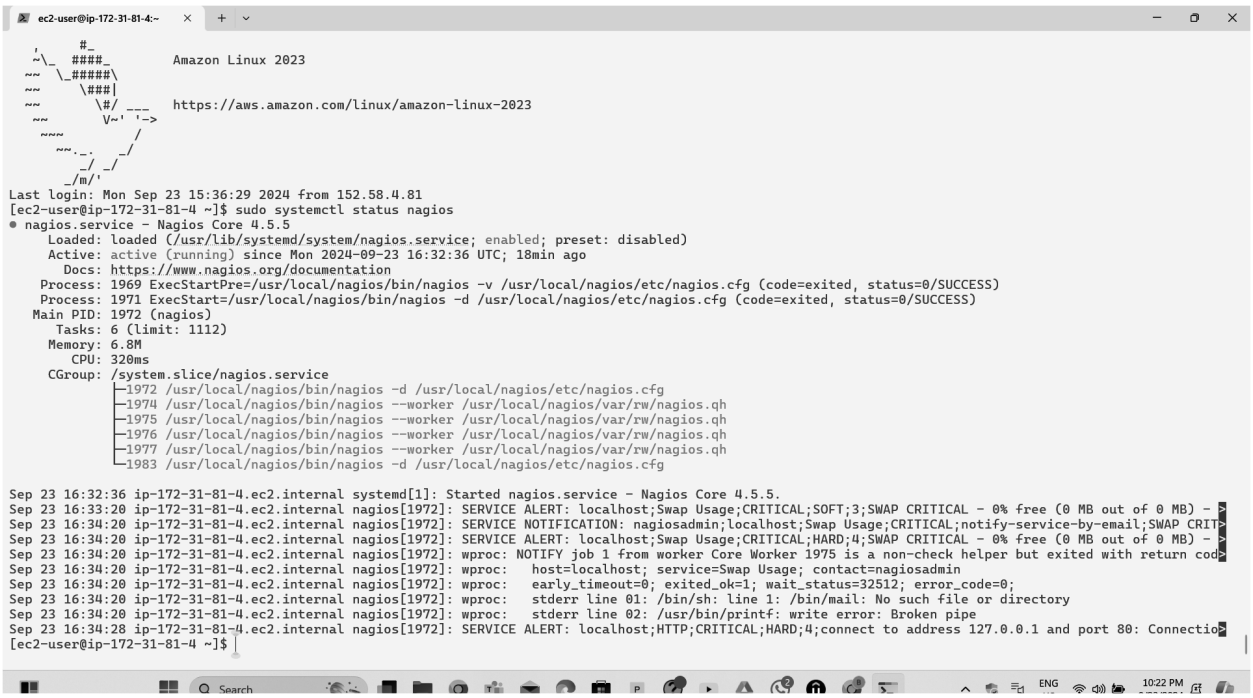
Port and Service Monitoring
Port and service monitoring in Nagios involves checking the availability and responsiveness of network services running on specific ports. This ensures that critical services (like HTTP, FTP, or SSH) are operational. Nagios uses plugins to ping the ports and verify whether services are up and responding as expected, allowing administrators to be alerted in case of outages.

Windows/Linux Server Monitoring
Windows/Linux server monitoring with Nagios entails tracking the performance and health of servers running these operating systems. It includes monitoring metrics such as CPU usage, memory consumption, disk space, and system logs. Nagios employs various plugins to gather data, enabling administrators to ensure optimal performance, identify potential issues, and maintain uptime across their server infrastructure.

Prerequisites:
AWS Academy or Personal account.
Nagios Server running on Amazon Linux Machine. (Refer Experiment No 9)

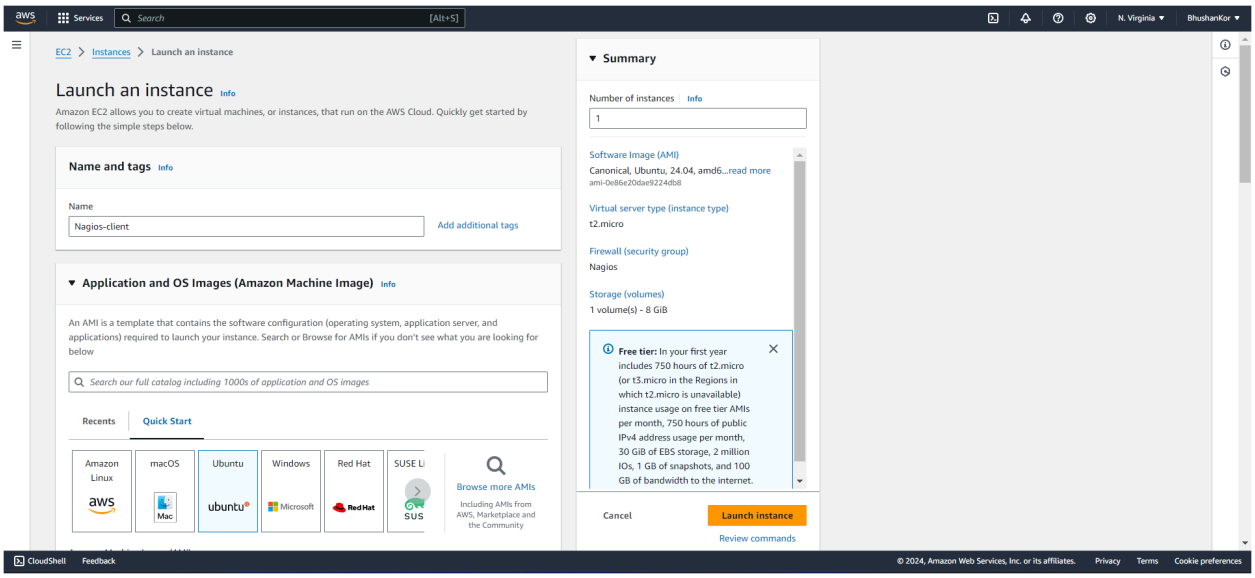
Monitoring Using Nagios:

Step 1: To Confirm Nagios is running on the server side Perform the following command on your Amazon Linux Machine (Nagios-host).
sudo systemctl status nagios



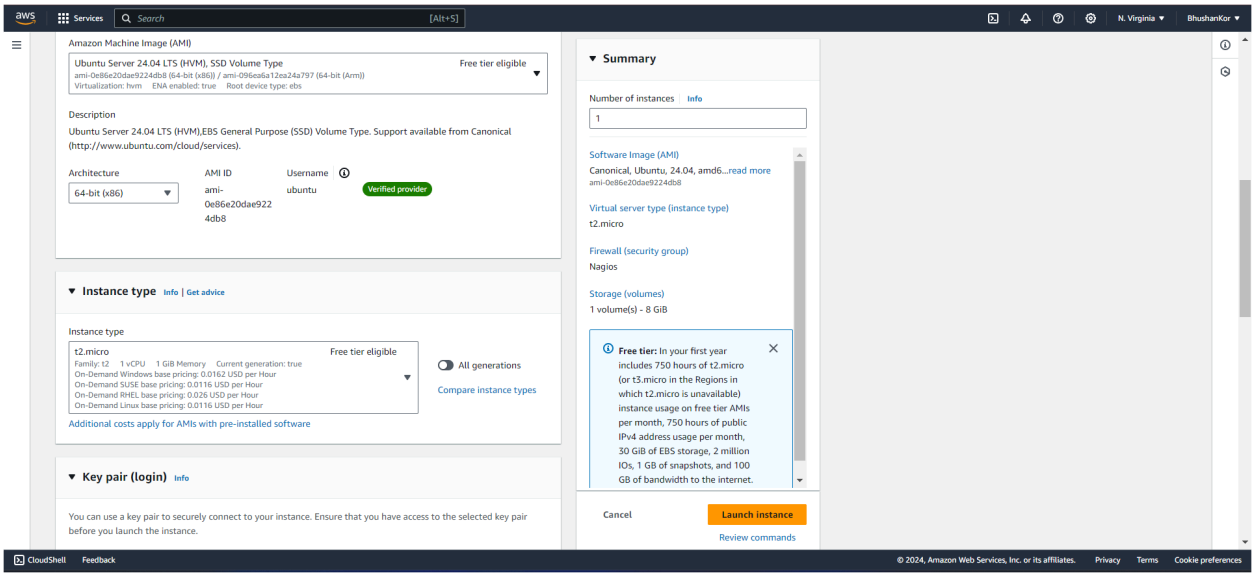
You can now proceed if you get the above message/output.

Step 2: Now Create a new EC2 instance. Name: Nagios-client, AMI: Ubuntu Instance Type: t2.micro.



For Key pair : Click on create key and make key of type RSA with extension .pem . Key will be downloaded to your local machine.

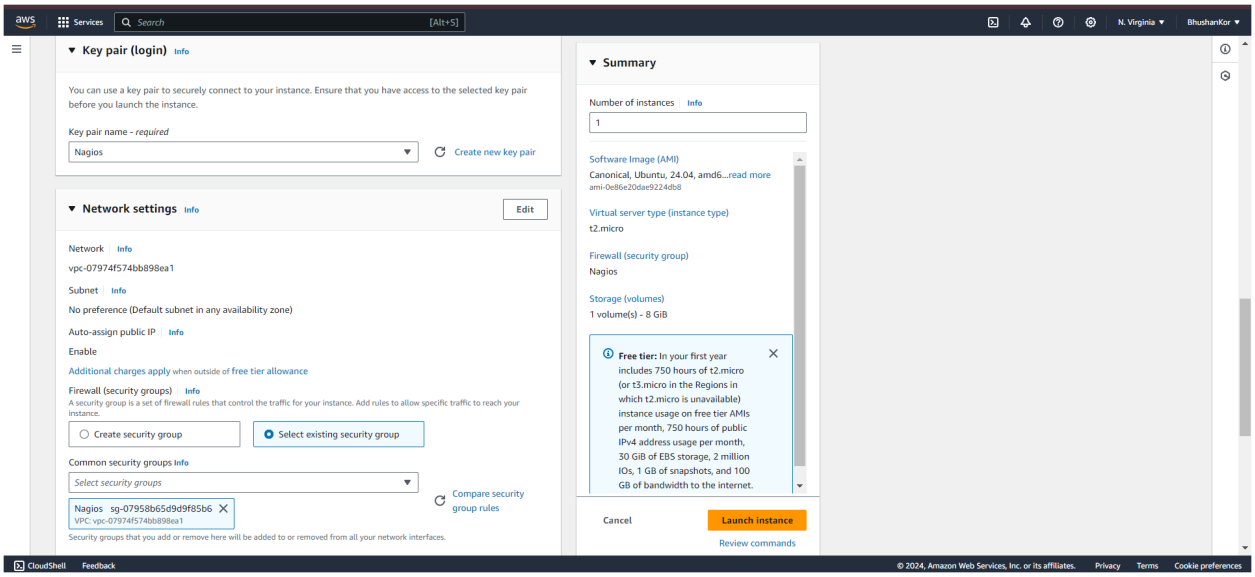
Now select that key in key pair if you already have key with type RSA and extension .pem no need to create new key but you must have that key downloaded.



Select the Existing Security Group and select the Security Group that we have created in Experiment no 9 or the same one you have used for the Nagios server (Nagios-host).

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

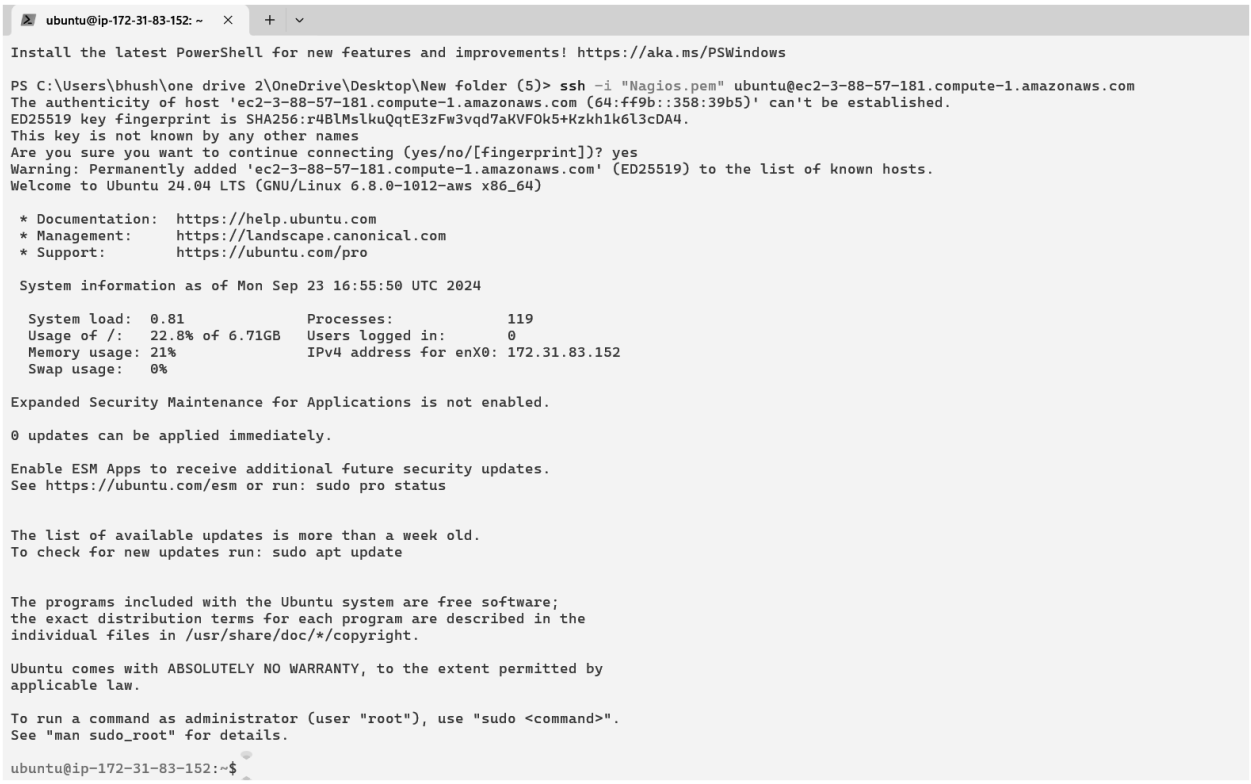


Step 3: Now After creating the EC2 Instance click on connect and then copy the command which is given as example in the SSH Client section .
Now open the terminal in the folder where your key(RSA key with .pem) is located. and paste that copied command.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Successfully connected to the instance.



Now perform all the commands on the Nagios-host till step 10

Step 4: Now on the server Nagios-host run the following command.

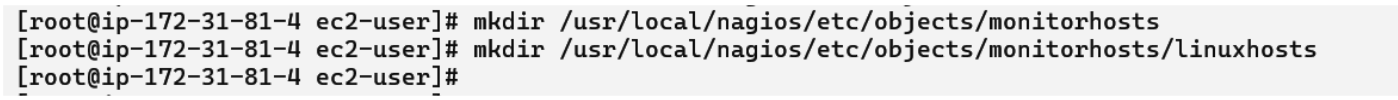
ps -ef | grep nagios

Step 5: Now Become root user and create root directories.

sudo su

mkdir /usr/local/nagios/etc/objects/monitorhosts

mkdir /usr/local/nagios/etc/objects/monitorhosts/linuxhosts



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 6: Copy the sample localhost.cfg to linuxhost.cfg by running the following command.(Below command should come in one line see screenshot below)

cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg

```
[root@ip-172-31-81-4 ec2-user]# cp /usr/local/nagios/etc/objects/localhost.cfg /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg
[root@ip-172-31-81-4 ec2-user]#
```

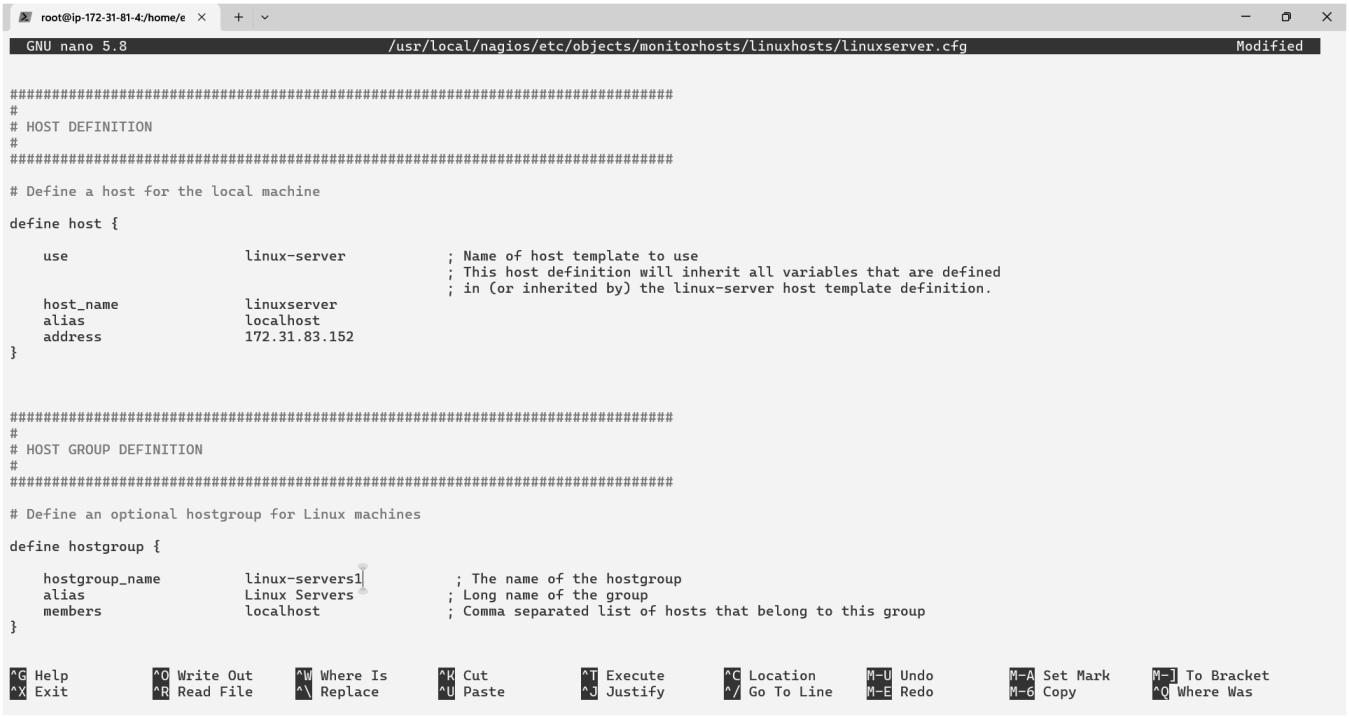
Step 7: Open linuxserver.cfg using nano and make the following changes in all positions?everywhere in file.

Change `hostname` to **linuxserver**.

Change `address` to the public IP of your Linux client.

Set `hostgroup_name` to **linux-servers1**.

nano /usr/local/nagios/etc/objects/monitorhosts/linuxhosts/linuxserver.cfg



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 8: Now update the Nagios config file .Add the following line in the file.

Line to add : cfg_dir=/usr/local/nagios/etc/objects/monitorhosts/
Run the command : **nano /usr/local/nagios/etc/nagios.cfg**

Step 9: Now Verify the configuration files by running the following commands.
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

```
[root@ip-172-31-81-4 ec2-user]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.5.5
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2024-09-17
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
        Checked 16 services.
        Checked 2 hosts.
        Checked 2 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 2 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
[root@ip-172-31-81-4 ec2-user]#
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 10: Now restart the services of nagios by running the following command.
service nagios restart

```
[root@ip-172-31-81-4 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-81-4 ec2-user]# sudo systemctl status nagi
```

Step 11: Now Go to the Nagios-client ssh terminal and update and install the packages by running the following command.
sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins

```
ubuntu@ip-172-31-83-152:~$ sudo apt update -y
sudo apt install gcc -y
sudo apt install -y nagios-nrpe-server nagios-plugins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
135 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  cpp cpp-13 cpp-13-x86-64-linux-gnu cpp-x86-64-linux-gnu gcc-13 gcc-13-base gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu libasan8 libatomic1 libcc1-0
  libgcc-13-dev libgomp1 libhwasan0 libisl23 libitm1 liblsan0 libmpc3 libquadmath0 libtsan2 libubsan1
Suggested packages:
  cpp-doc gcc-13-locales cpp-13-doc gcc-multilib make autoconf automake libtool flex bison gdb gcc-doc gcc-13-multilib gcc-13-doc gdb-x86-64-linux-gnu
The following NEW packages will be installed:
  cpp cpp-13 cpp-13-x86-64-linux-gnu cpp-x86-64-linux-gnu gcc gcc-13 gcc-13-base gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu libasan8 libatomic1 libcc1-0
  libgcc-13-dev libgomp1 libhwasan0 libisl23 libitm1 liblsan0 libmpc3 libquadmath0 libtsan2 libubsan1
0 upgraded, 22 newly installed, 0 to remove and 135 not upgraded.
Need to get 47.3 MB of archives.
After this operation, 163 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 gcc-13-base amd64 13.2.0-23ubuntu4 [49.0 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libisl23 amd64 0.26-3build1 [680 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libmpc3 amd64 1.3.1-1build1 [54.5 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 cpp-13-x86-64-linux-gnu amd64 13.2.0-23ubuntu4 [11.2 MB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 cpp-13 amd64 13.2.0-23ubuntu4 [1032 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 cpp-x86-64-linux-gnu amd64 4:13.2.0-7ubuntu1 [5326 B]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 cpp amd64 4:13.2.0-7ubuntu1 [22.4 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libcc1-0 amd64 14-20240412-0ubuntu1 [47.7 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libgomp1 amd64 14-20240412-0ubuntu1 [147 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libitm1 amd64 14-20240412-0ubuntu1 [28.9 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 libatomic1 amd64 14-20240412-0ubuntu1 [10.4 kB]
```



```
ubuntu@ip-172-31-83-152: ~
monitoring-plugins is already the newest version (2.3.5-1ubuntu3).
Suggested packages:
  xinetd | inetd
The following NEW packages will be installed:
  nagios-nrpe-server
0 upgraded, 1 newly installed, 0 to remove and 135 not upgraded.
Need to get 356 kB of archives.
After this operation, 469 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 nagios-nrpe-server amd64 4.1.0-1ubuntu3 [356 kB]
Fetched 356 kB in 0s (15.6 MB/s)
Selecting previously unselected package nagios-nrpe-server.
(Reading database ... 73771 files and directories currently installed.)
Preparing to unpack .../nagios-nrpe-server_4.1.0-1ubuntu3_amd64.deb ...
Unpacking nagios-nrpe-server (4.1.0-1ubuntu3) ...
Setting up nagios-nrpe-server (4.1.0-1ubuntu3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...

Service restarts being deferred:
 /etc/needrestart/restart.d/dbus.service
 systemctl restart getty@tty1.service
 systemctl restart networkd-dispatcher.service
 systemctl restart serial-getty@ttyS0.service
 systemctl restart systemd-logind.service
 systemctl restart unattended-upgrades.service

No containers need to be restarted.

User sessions running outdated binaries:
 ubuntu @ session #2: sshd[1001,1111]
 ubuntu @ session #8: sshd[1205,1263]
 ubuntu @ user manager service: systemd[1006]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-83-152:~$
```

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 12: Open nrpe.cfg file to make changes.Under allowed_hosts, add your nagios host IP address.
sudo nano /etc/nagios/nrpe.cfg

Step 13: Now restart the NRPE server by this command.
sudo systemctl restart nagios-nrpe-server

```
ubuntu@ip-172-31-83-152:~$ sudo nano /etc/nagios/nrpe.cfg

ubuntu@ip-172-31-83-152:~$ sudo systemctl restart nagios-nrpe-server
ubuntu@ip-172-31-83-152:~$
```

Step 14: Now again check the status of Nagios by running this command on Nagios-host and also check httpd is active and run the command to active it.

sudo systemctl status nagios

```
[root@ip-172-31-81-4 ec2-user]# service nagios restart
Redirecting to /bin/systemctl restart nagios.service
[root@ip-172-31-81-4 ec2-user]# sudo systemctl status nagios
● nagios.service - Nagios Core 4.5.5
     Loaded: loaded (/usr/lib/systemd/system/nagios.service; enabled; preset: disabled)
     Active: active (running) since Mon 2024-09-23 17:26:10 UTC; 1min 39s ago
       Docs: https://www.nagios.org/documentation
    Process: 4227 ExecStartPre=/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
    Process: 4228 ExecStart=/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg (code=exited, status=0/SUCCESS)
   Main PID: 4234 (nagios)
      Tasks: 6 (limit: 1112)
     Memory: 4.2M
        CPU: 60ms
     CGroup: /system.slice/nagios.service
             ├─4234 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
             ├─4236 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─4237 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─4238 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             ├─4239 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
             └─4242 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: qh: echo service query handler registered
Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: qh: help for the query handler registered
Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: wproc: Successfully registered manager as @wproc with query handler
Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: wproc: Registry request: name=Core Worker 4238;pid=4238
Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: wproc: Registry request: name=Core Worker 4239;pid=4239
Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: wproc: Registry request: name=Core Worker 4237;pid=4237
Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: wproc: Registry request: name=Core Worker 4236;pid=4236
Sep 23 17:26:10 ip-172-31-81-4.ec2.internal nagios[4234]: Successfully launched command file worker with pid 4242
Sep 23 17:27:00 ip-172-31-81-4.ec2.internal nagios[4234]: SERVICE ALERT: linuxserver;HTTP;CRITICAL;SOFT;3;connect to address 172.31.83.152 and port 80: Con>
Sep 23 17:27:30 ip-172-31-81-4.ec2.internal nagios[4234]: SERVICE ALERT: linuxserver;Swap Usage;CRITICAL;SOFT;1;SWAP CRITICAL - 0% free (0 MB out of 0 MB) >
[root@ip-172-31-81-4 ec2-user]#
```

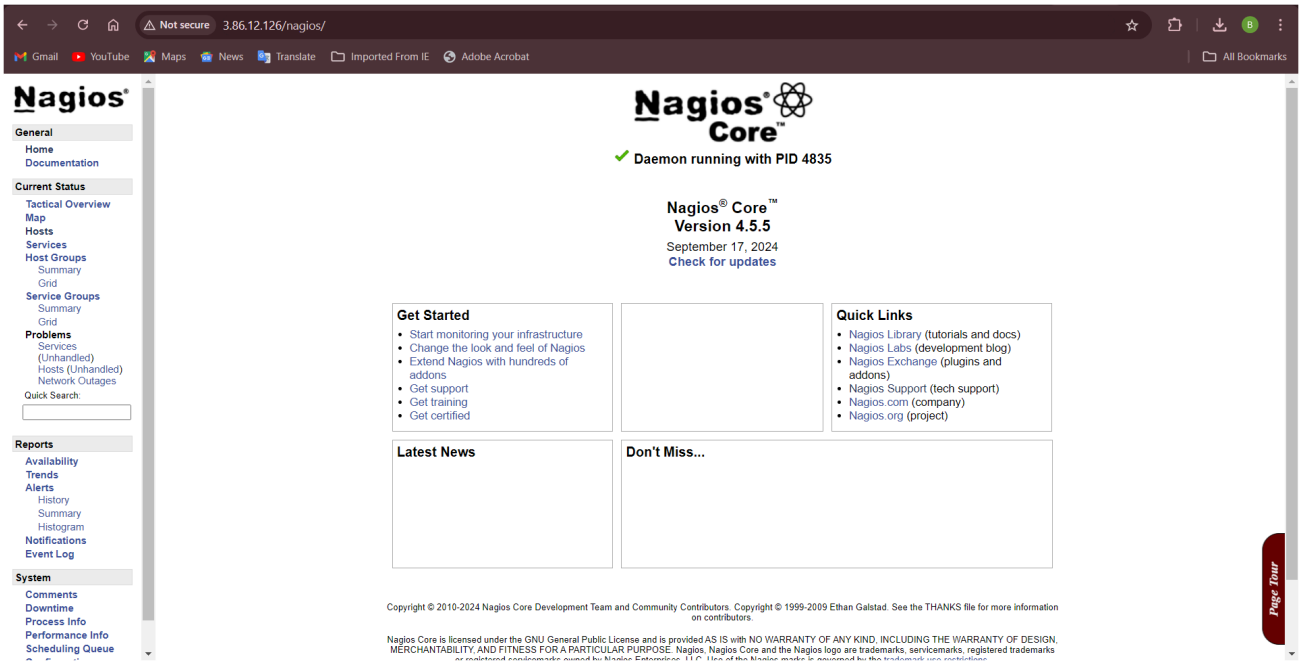
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

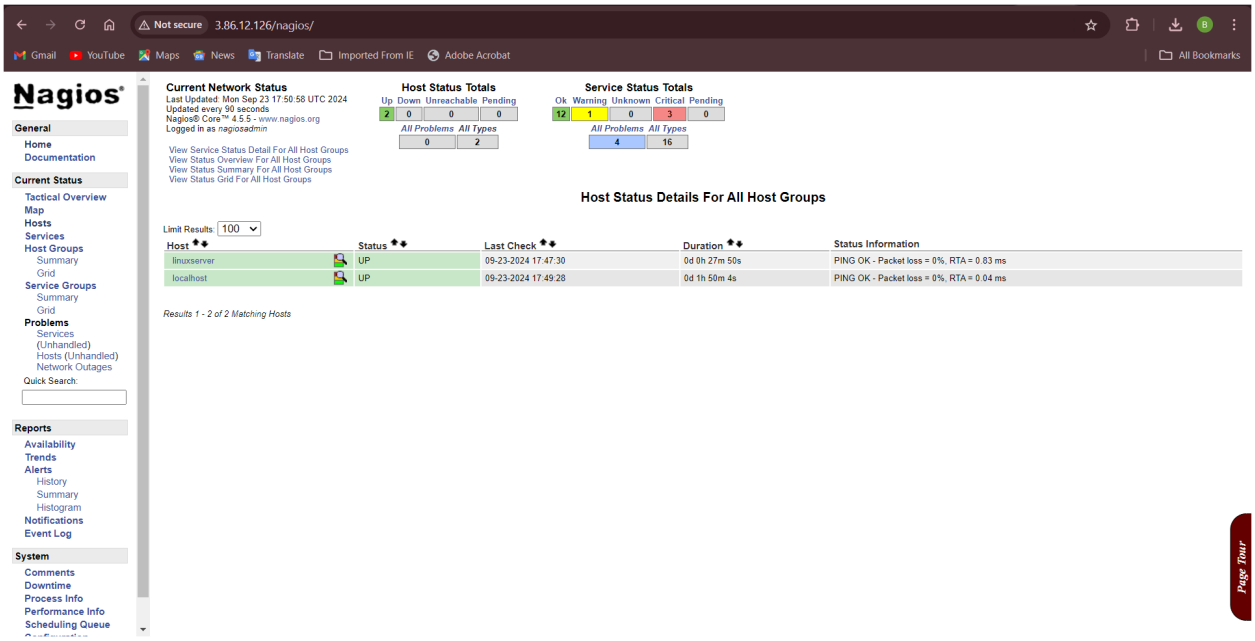
sudo systemctl status httpd
sudo systemctl start httpd
sudo systemctl enable httpd

```
[root@ip-172-31-81-4 ec2-user]# sudo systemctl status httpd
○ httpd.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
    Drop-In: /usr/lib/systemd/system/httpd.service.d
             └─php-fpm.conf
     Active: inactive (dead)
       Docs: man:httpd.service(8)
[root@ip-172-31-81-4 ec2-user]# sudo systemctl start httpd
[root@ip-172-31-81-4 ec2-user]# sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[root@ip-172-31-81-4 ec2-user]#
```

Step 15: Now to check Nagios dashboard go to http://<Nagios-host ip>/nagios .



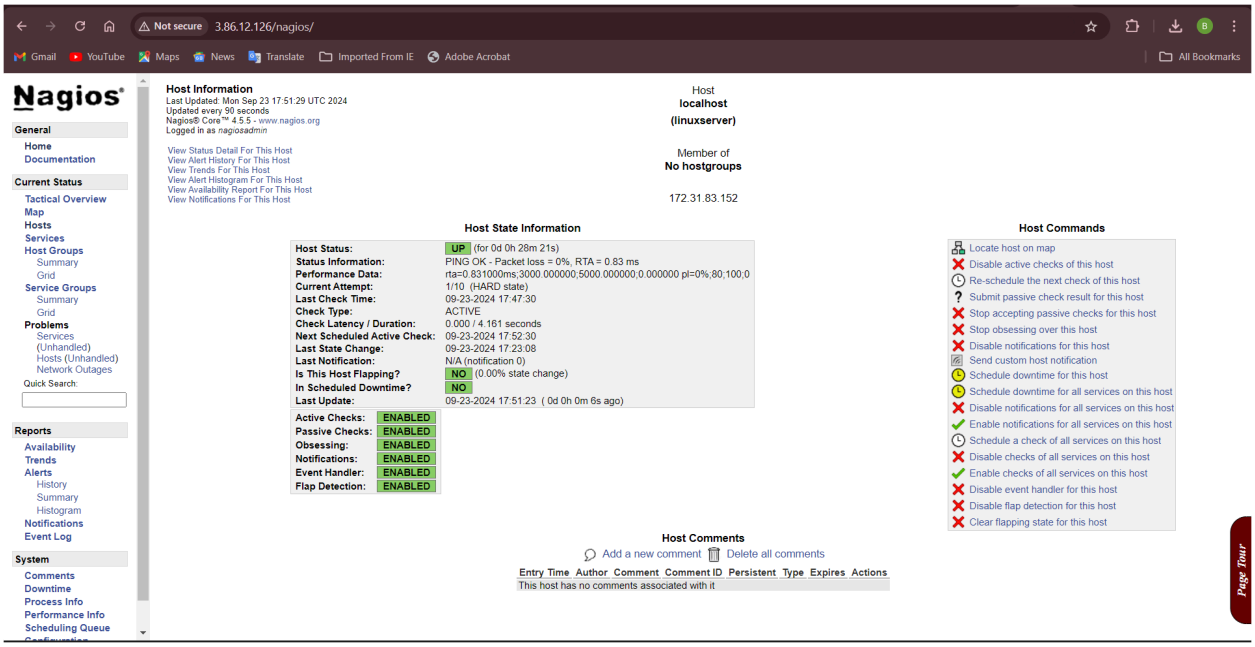
Now Click on Hosts from left side panel



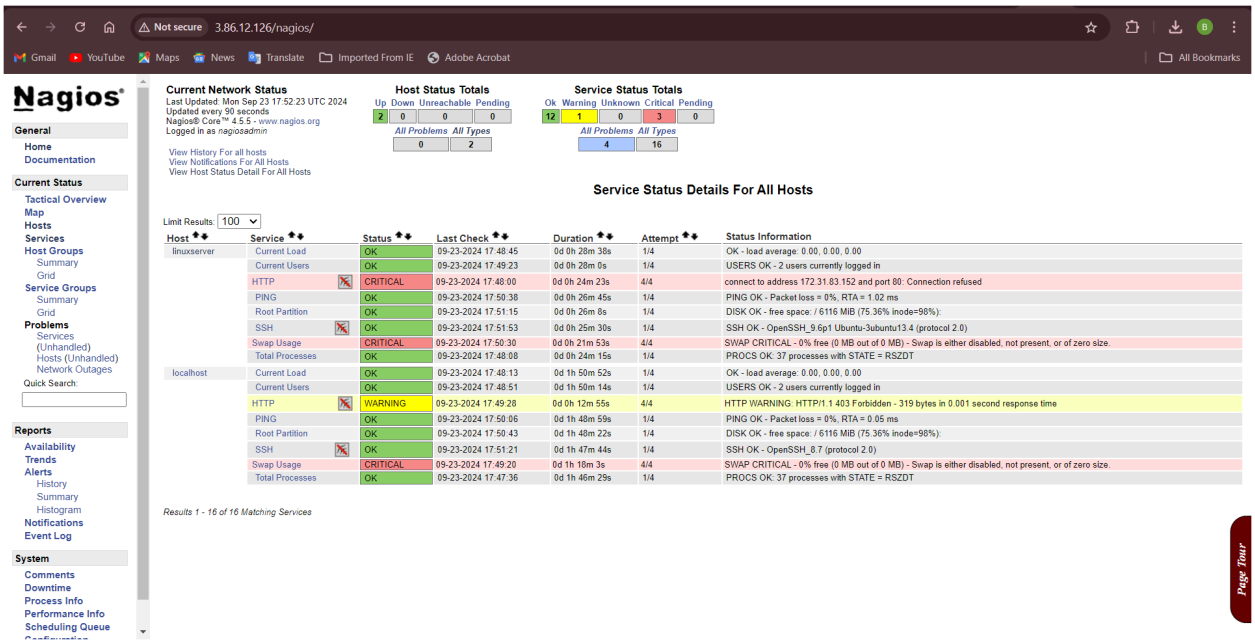
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

We can see our linuxserver now click on it we can see the host information.



Current Network Status



Conclusion: In conclusion, the experiment focused on monitoring ports, services, and a Linux server using Nagios. Through the step-by-step process, we successfully configured Nagios to monitor essential network services on the Linux server. By setting up both the Nagios host and client, we were able to track system performance, ensure service availability, and monitor key metrics like CPU and memory usage.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

Theory:

AWS Lambda

A fully managed, serverless computing service where you run code without provisioning or managing servers. Lambda automatically scales your application based on the number of incoming requests or events, ensuring efficient resource utilization. You are only charged for the time your code is running, with no upfront cost, making it cost-effective for on-demand workloads.

Lambda Workflow

- **Create a Function**: Write the function code and define its handler (entry point). You can use the AWS Console, CLI, or upload a deployment package.
- **Set Event Sources**: Define how the function is triggered (e.g., when an object is uploaded to S3 or a DynamoDB table is updated).
- **Execution**: When triggered, Lambda runs your function, executes the logic, and automatically scales to handle the incoming event volume.
- **Scaling and Concurrency**: Lambda scales automatically by launching more instances of the function to handle simultaneous invocations. There are also options for configuring **reserved concurrency** to manage traffic.
- **Monitoring and Logging**: Lambda integrates with Amazon CloudWatch for logging and monitoring. Logs for each invocation are sent to CloudWatch, allowing you to track performance and troubleshoot errors.

AWS Lambda Functions

- **Python**: Great for quick development with its rich standard library and support for lightweight tasks.
- **Java**: Typically used for more complex, compute-intensive tasks. While it's robust, cold start times can be higher.
- **Node.js**: Excellent for I/O-bound tasks like handling APIs or streaming data, with fast startup times and efficient memory usage.

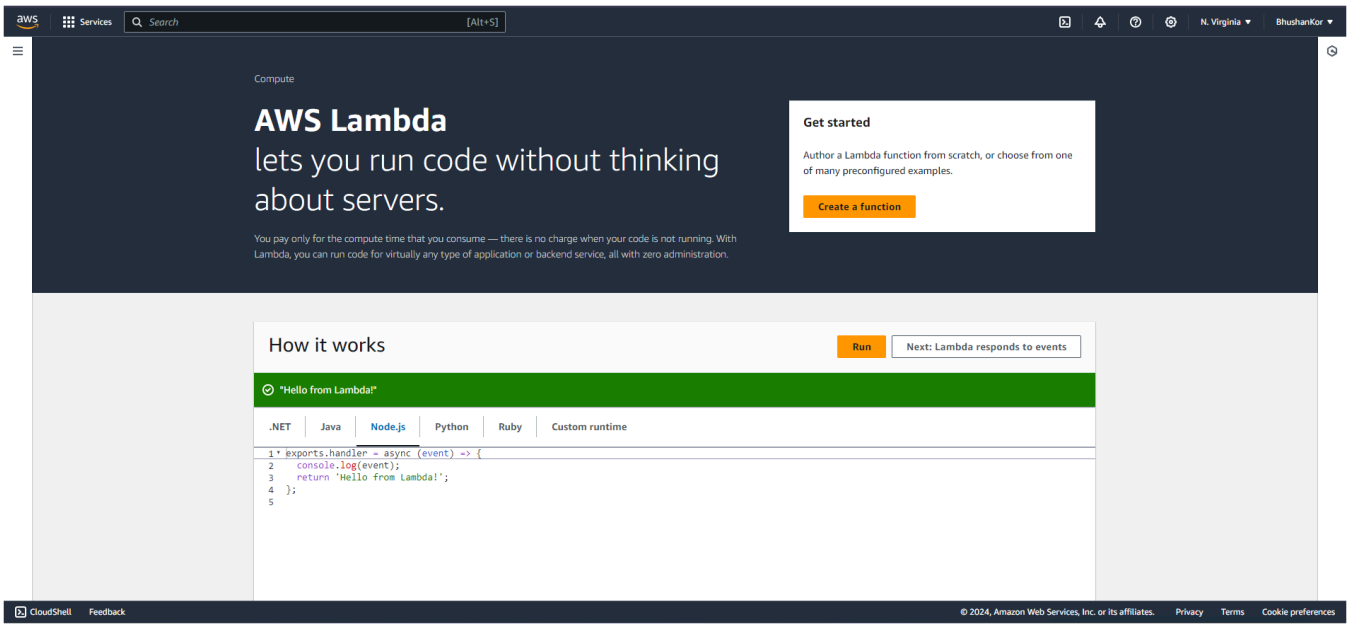
Prerequisites: AWS Personal/Academy Account

Name:Bhushan Mukund Kor **Academy Year:2024-2025**

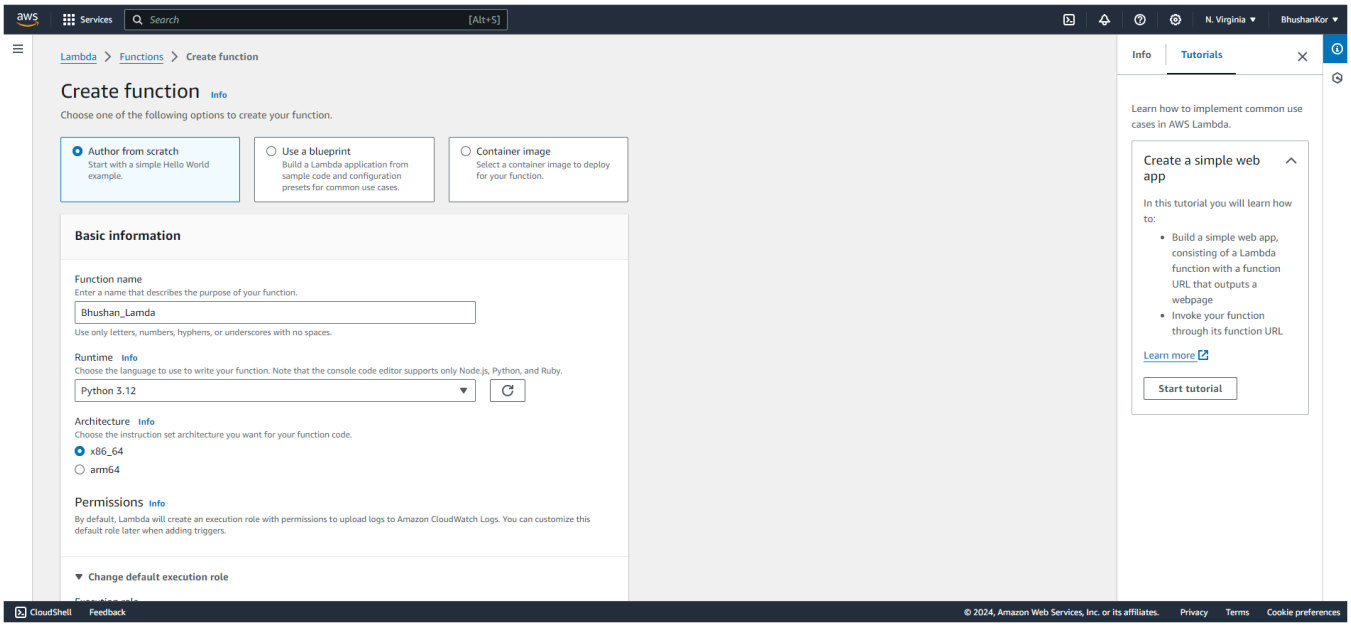
Division: D15C **Roll No: 28**

Steps To create the lambda function:

Step 1: Login to your AWS Personal/Academy Accout.Open lambda andclick on create function button.

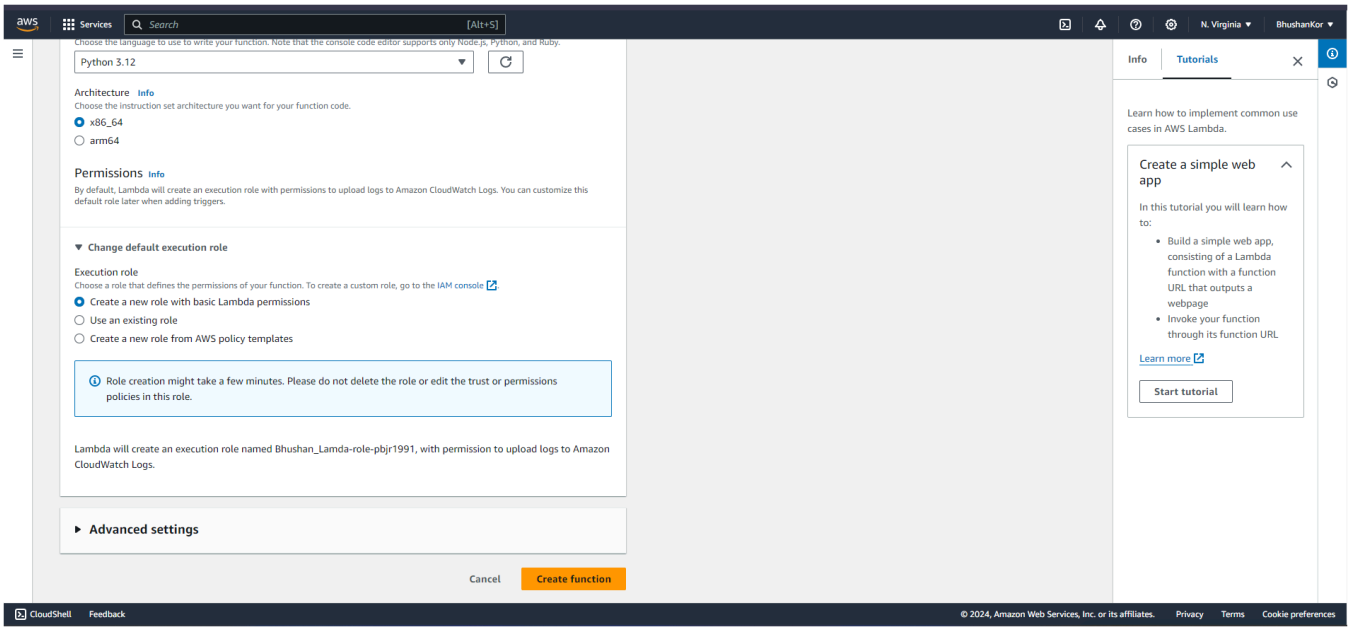


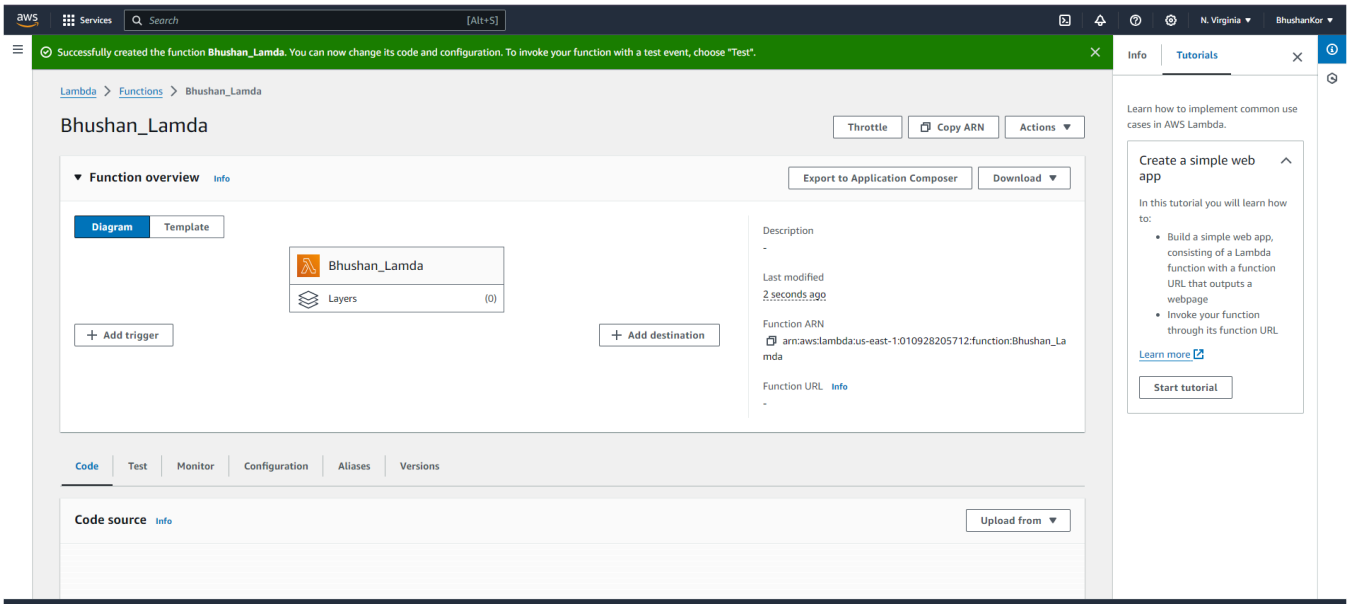
Step 2: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Exceution role to Create a new role with basic Lambda permissions.

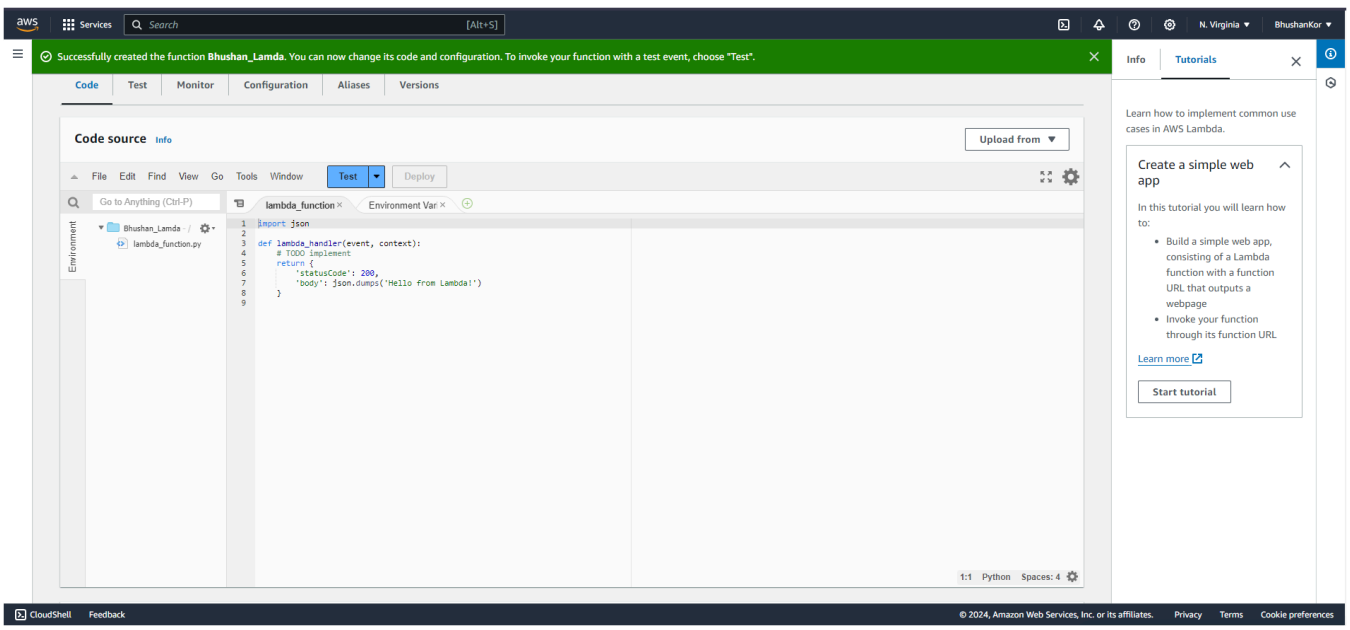


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



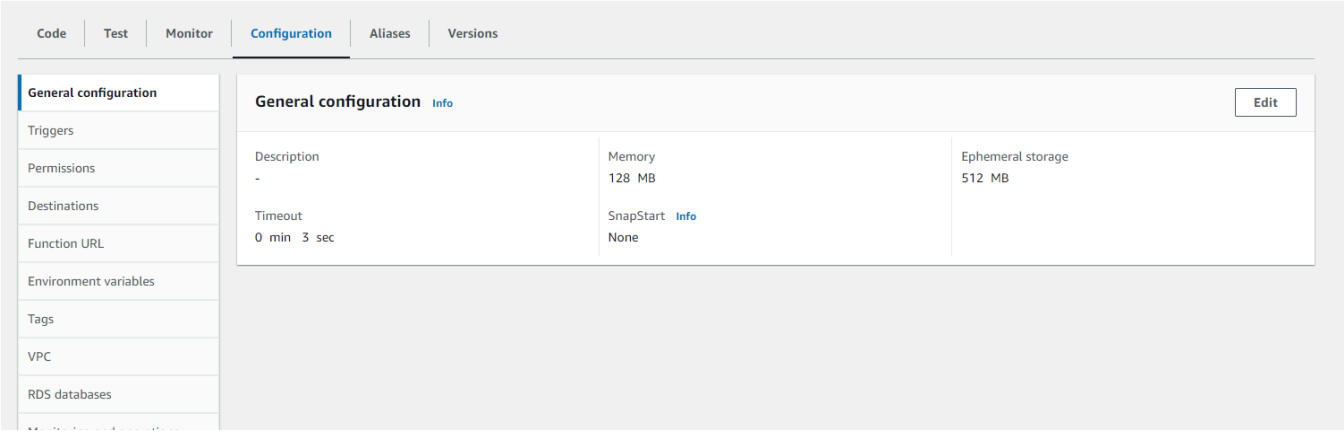




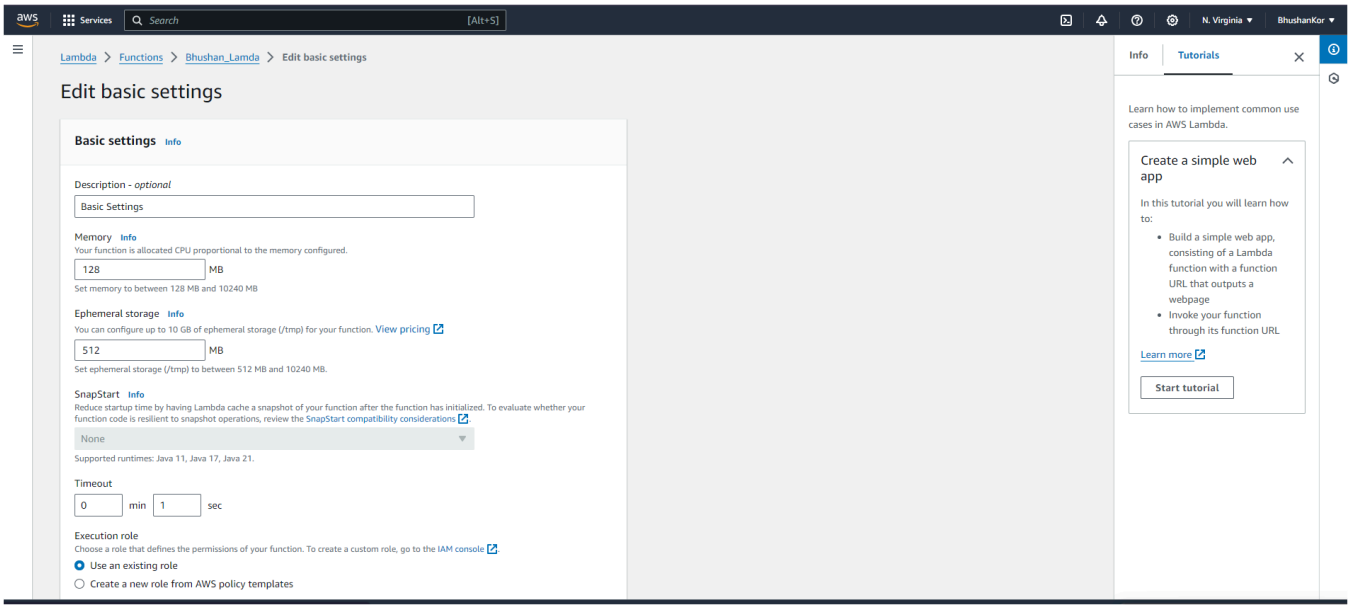
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

So See or Edit the basic settings go to configuration then click on edit general setting.



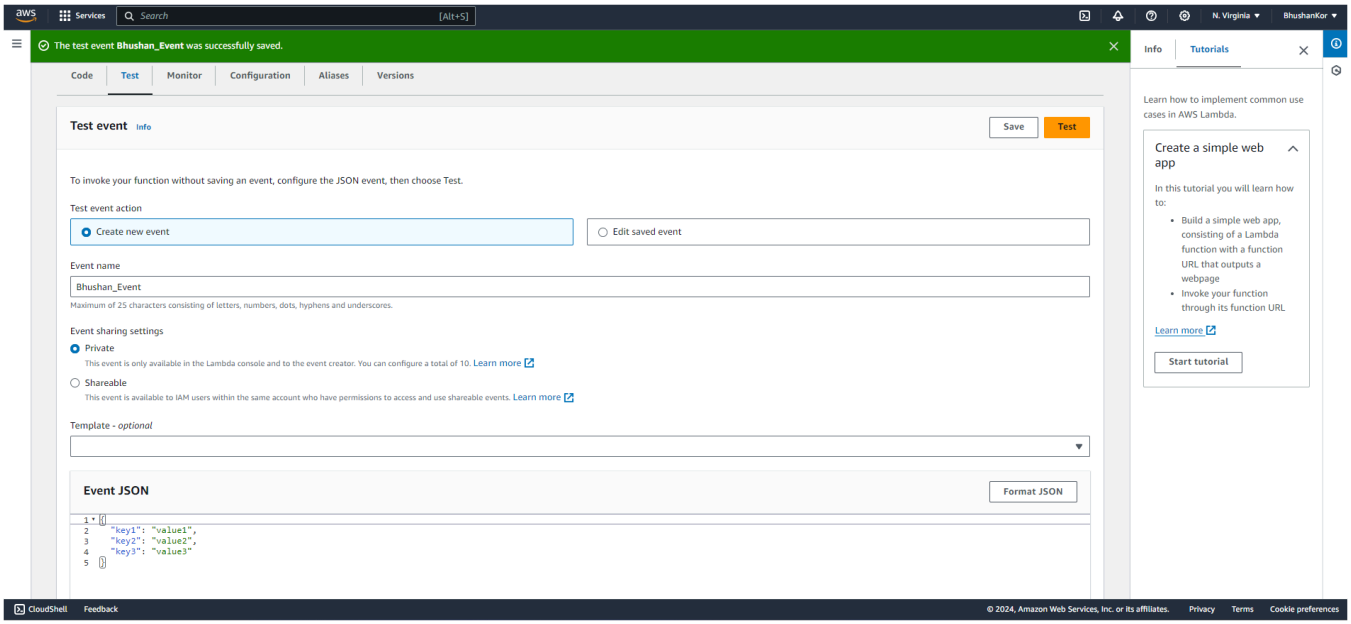
Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.



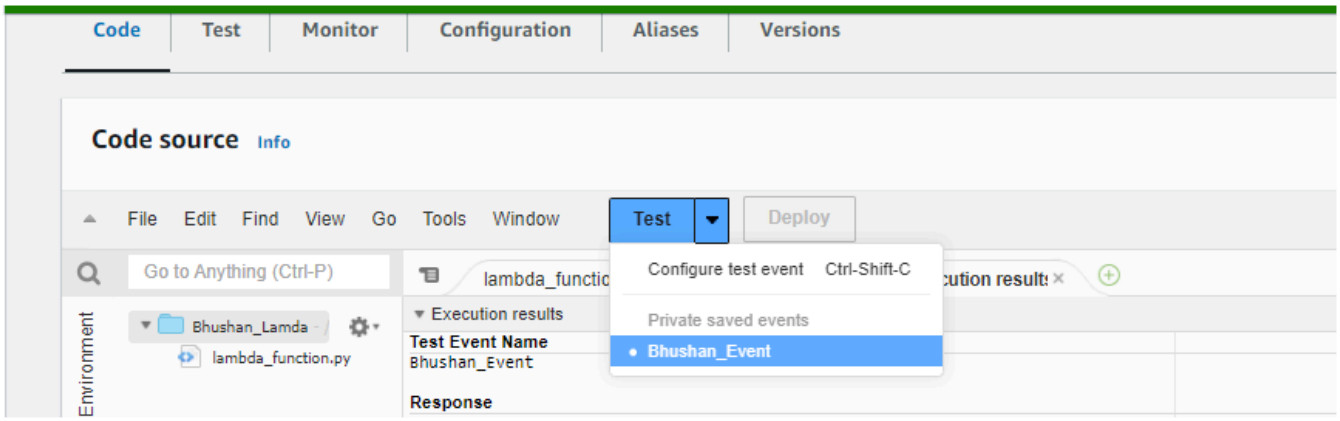
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

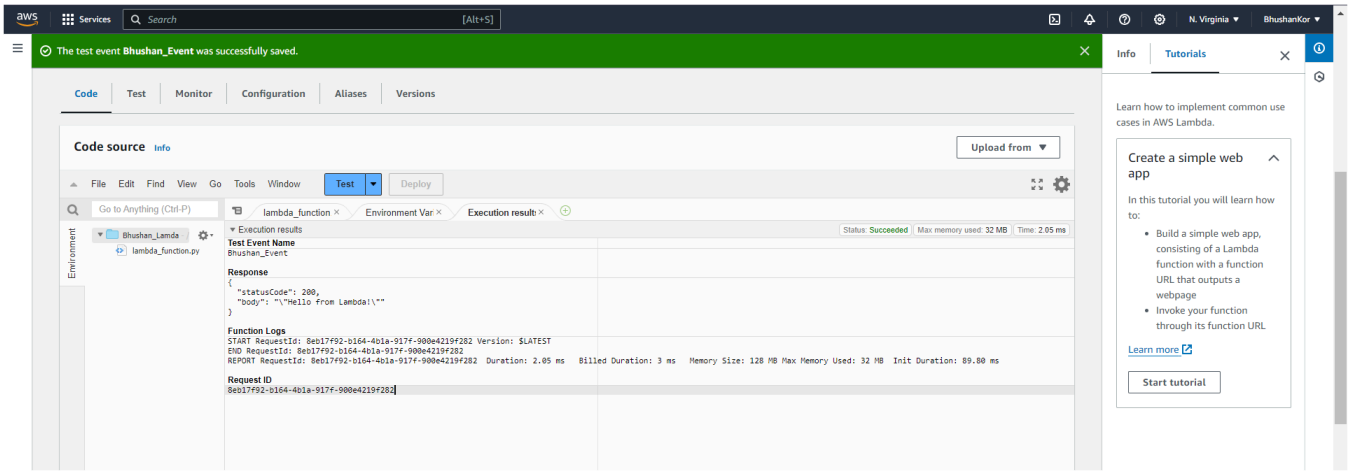
Division: D15C **Roll No: 28**

Step 3: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select hello-world template.



Step 4: Now In Code section select the created event from the dropdown of test then click on test . You will see the below output.

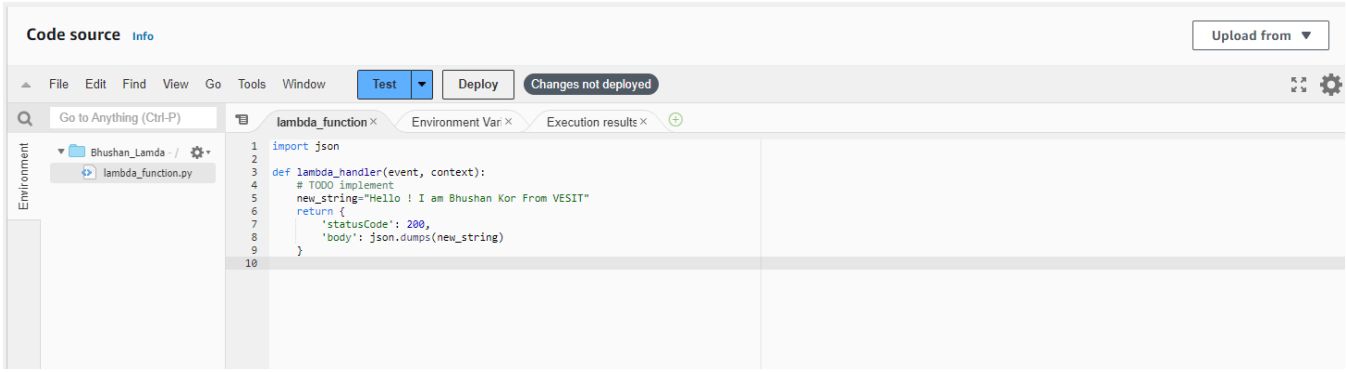




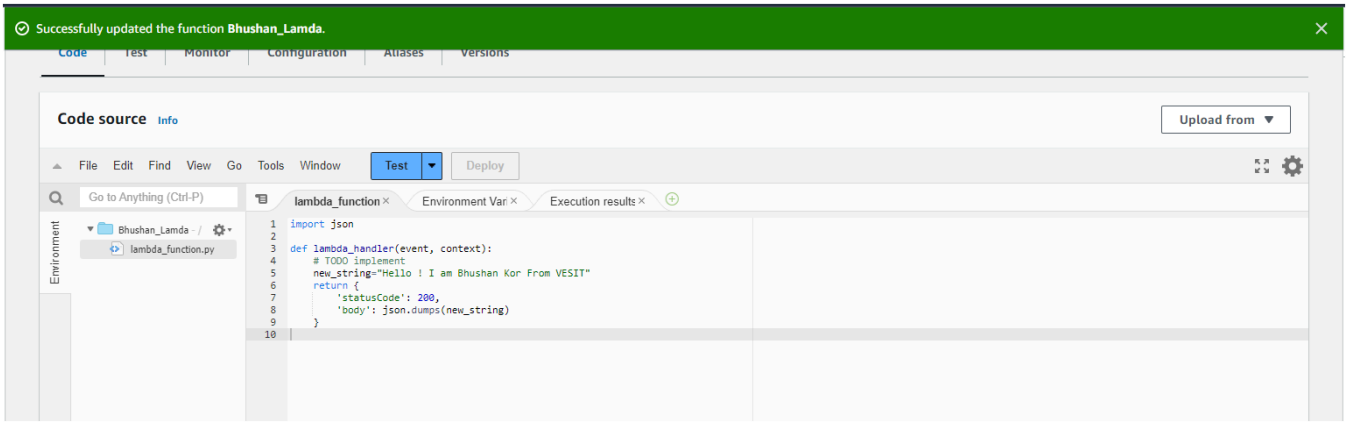
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

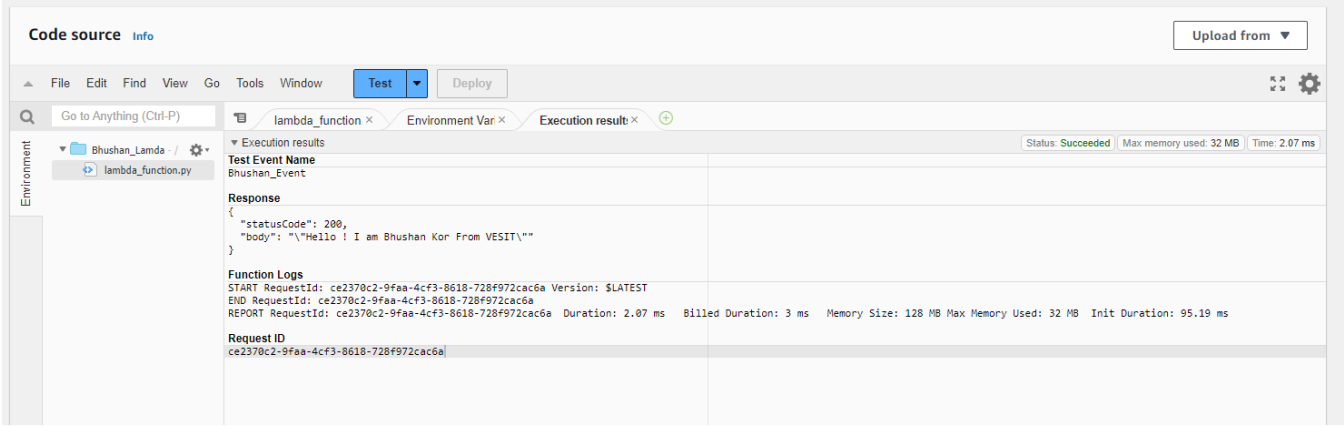
Step 5: You can edit your lambda function code. I have changed the code to display the new String.



Now ctrl+s to save and click on deploy to deploy the changes.



Step 6: Now click on the test and observe the output. We can see the status code 200 and your string output and function logs. On successful deployment.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Conclusion: In this experiment, we successfully created an AWS Lambda function and walked through its essential steps. After setting up the function with Python, we configured the basic settings, including adjusting the timeout to 1 second. We then created a test event, deployed the function, and validated the output. Additionally, we modified the Lambda function's code and redeployed it to observe the changes in real-time.

This practical experience demonstrated the simplicity and flexibility of AWS Lambda in creating serverless applications, allowing you to focus on code while AWS manages the infrastructure and scaling.

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Aim: To create a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3

Theory:

AWS Lambda and S3 Integration:

AWS Lambda allows you to execute code in response to various events, including those triggered by Amazon S3. When an object is added to an S3 bucket, it can trigger a Lambda function to execute, allowing for event-driven processing without managing servers.

Workflow:

1. Create an S3 Bucket:

○ First, create an S3 bucket that will store the objects. This bucket will act as the trigger source for the Lambda function.

2. Create the Lambda Function:

○ Set up a new Lambda function using AWS Lambda’s console. You can choose a runtime environment like Python, Node.js, or Java.

○ Write code that logs a message like “An Image has been added” when triggered.

3. Set Up Permissions:

○ Ensure that the Lambda function has the necessary permissions to access S3. You can do this by attaching an IAM role with policies that allow reading from the bucket and writing logs to CloudWatch.

4. Configure S3 Trigger:

○ Link the S3 bucket to the Lambda function by setting up a trigger. Specify that the function should be triggered when an object is created in the bucket (e.g., when an image is uploaded).

5. Test the Setup:

○ Upload an object (e.g., an image) to the S3 bucket to test the trigger. The Lambda function should execute and log the message “An Image has been added” in AWS CloudWatch Logs.

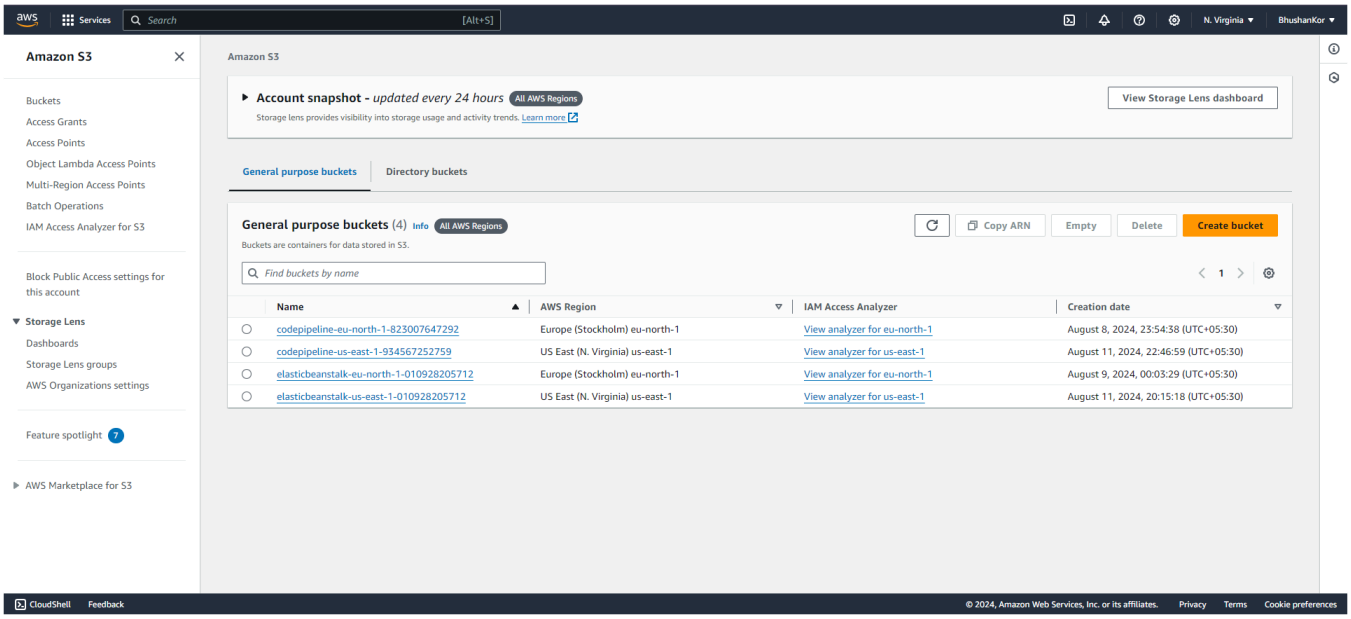
Prerequisites: AWS Personal Account

Name:Bhushan Mukund Kor **Academic Year:2024-2025**

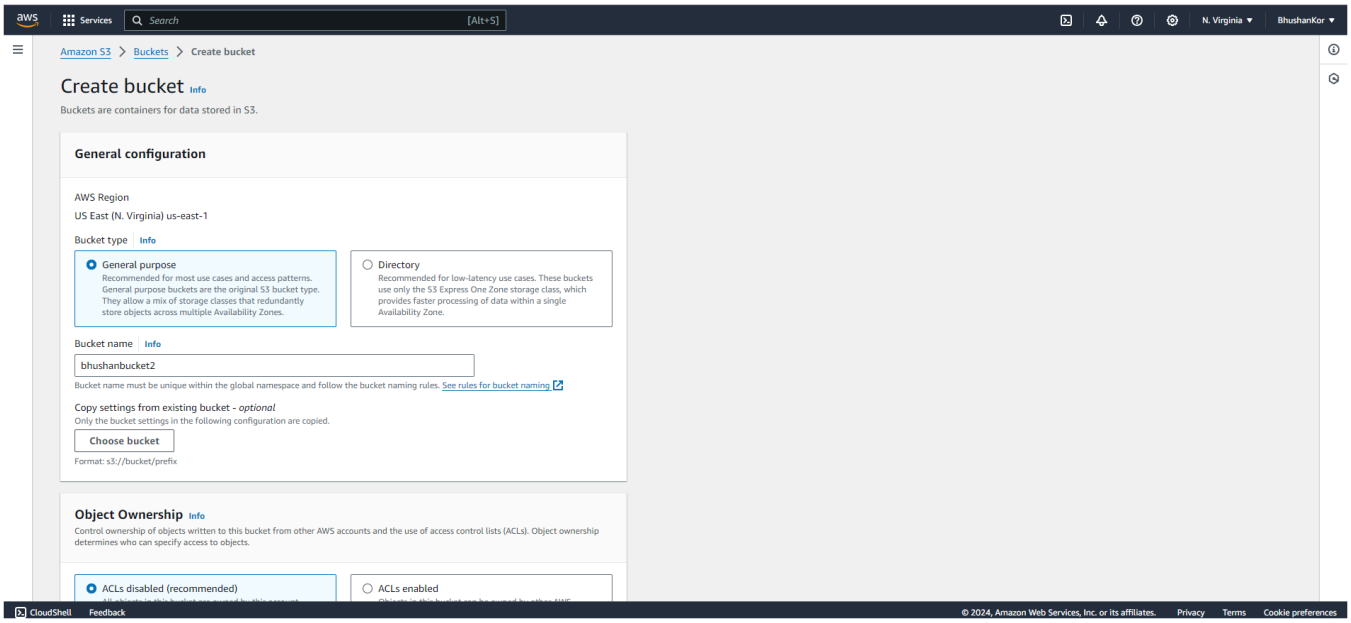
Division: D15C **Roll No: 28**

Steps To create the lambda function:

Step 1: Login to your AWS Personal account. Now open S3 from services and click on create S3 bucket.

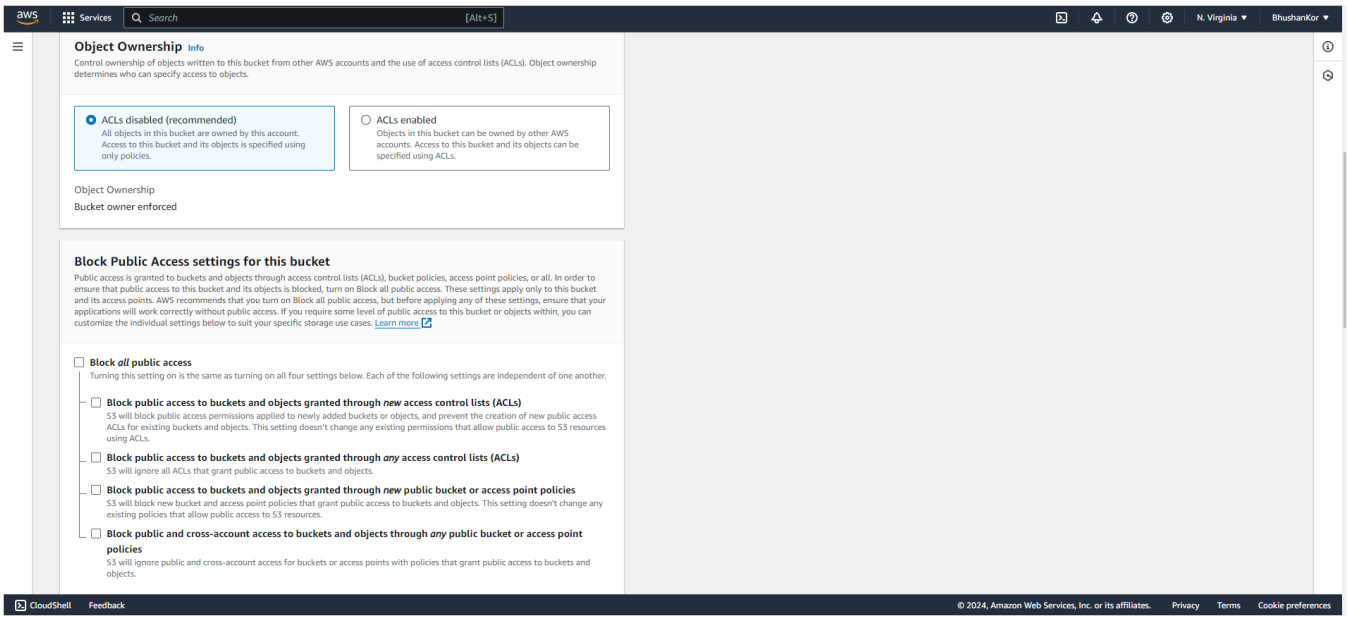


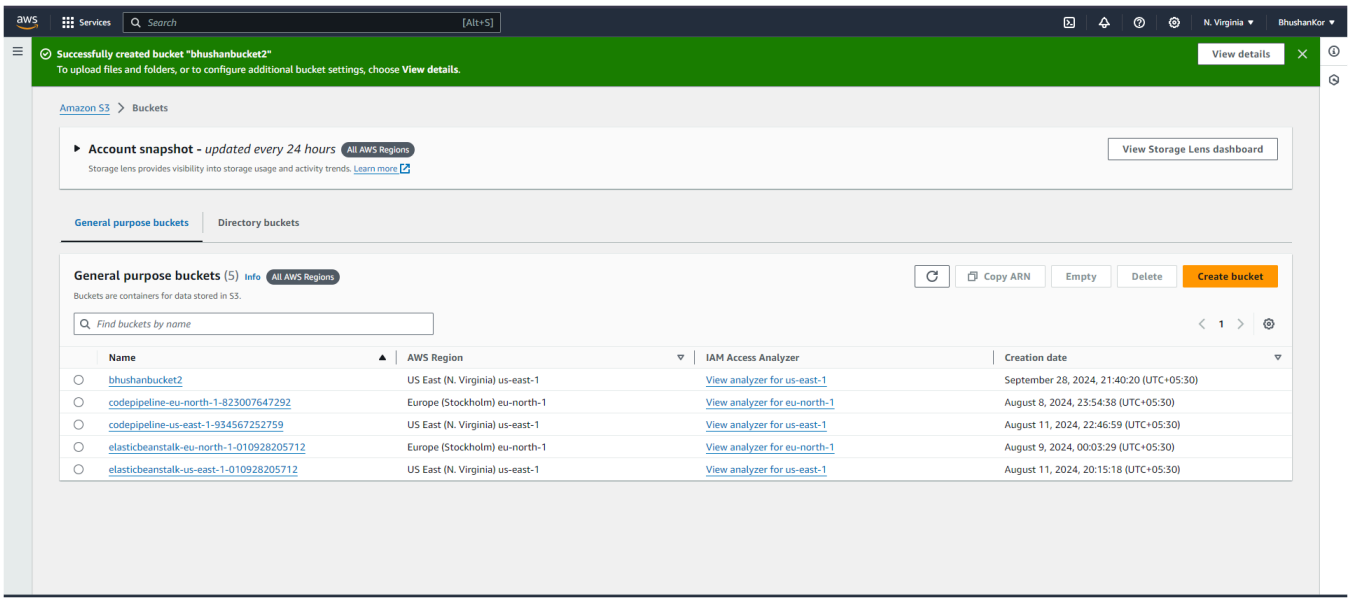
Step 2: Now Give a name to the Bucket, select general purpose project and deselect the Block public access and keep other this to default.



Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

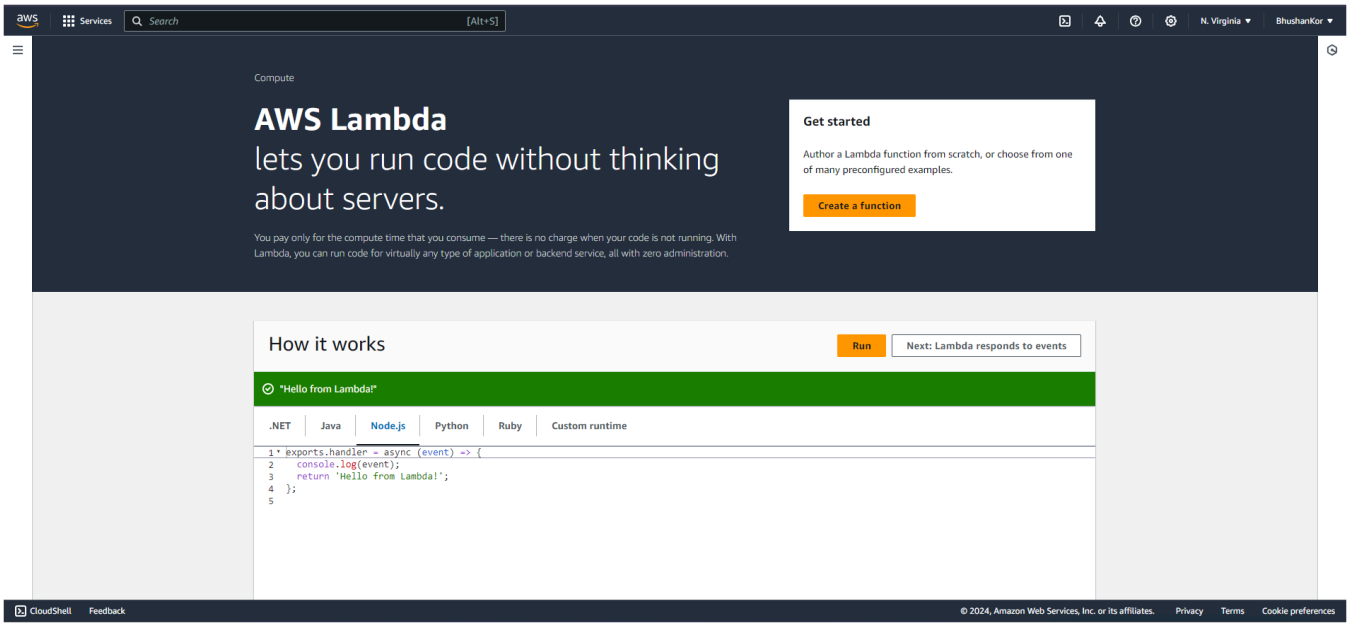




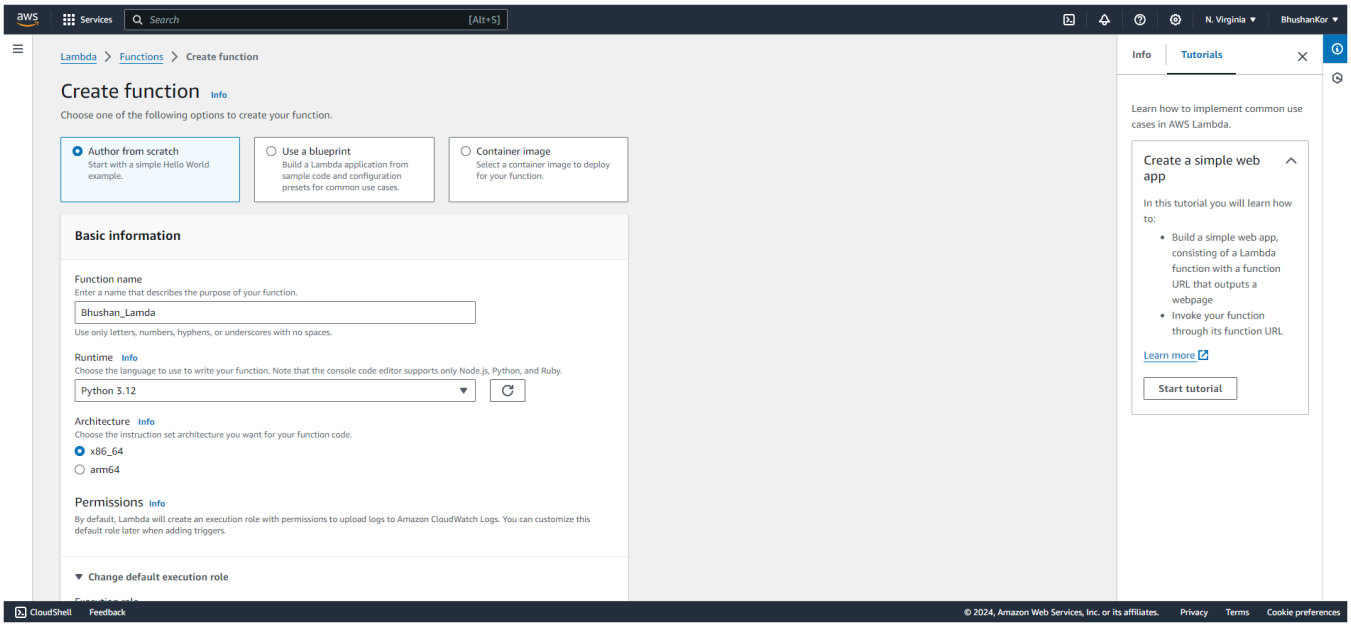
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 3: Open lambda console and click on create function button.

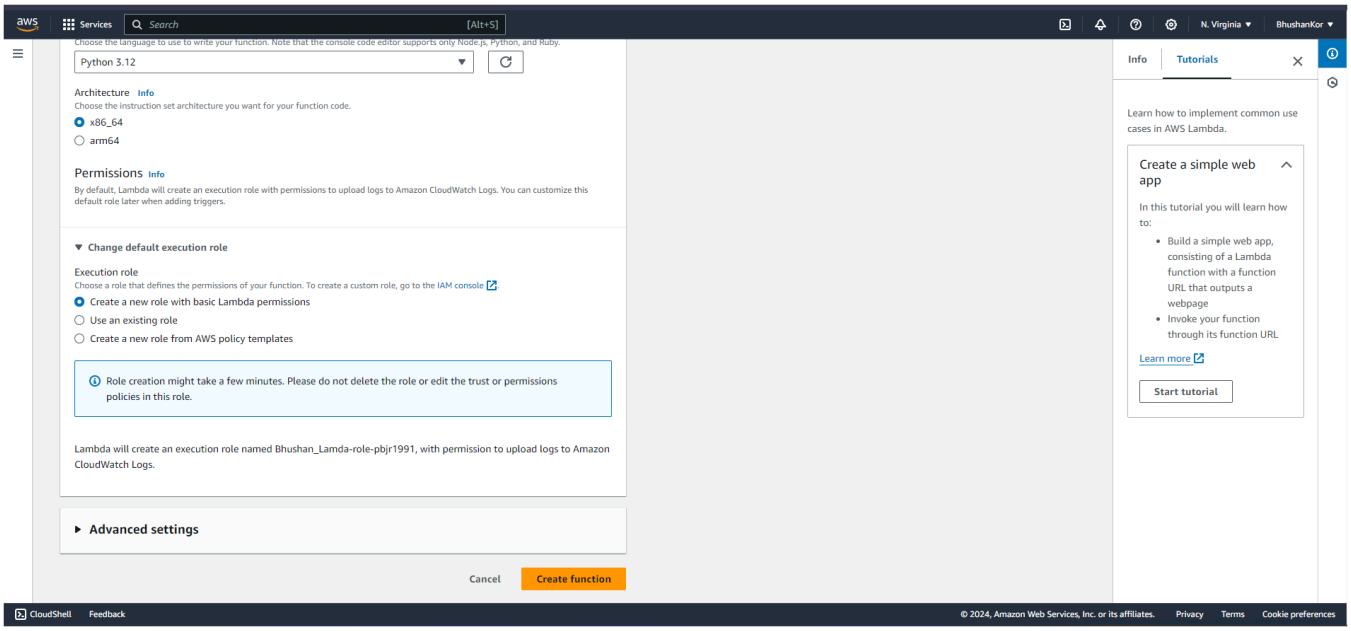


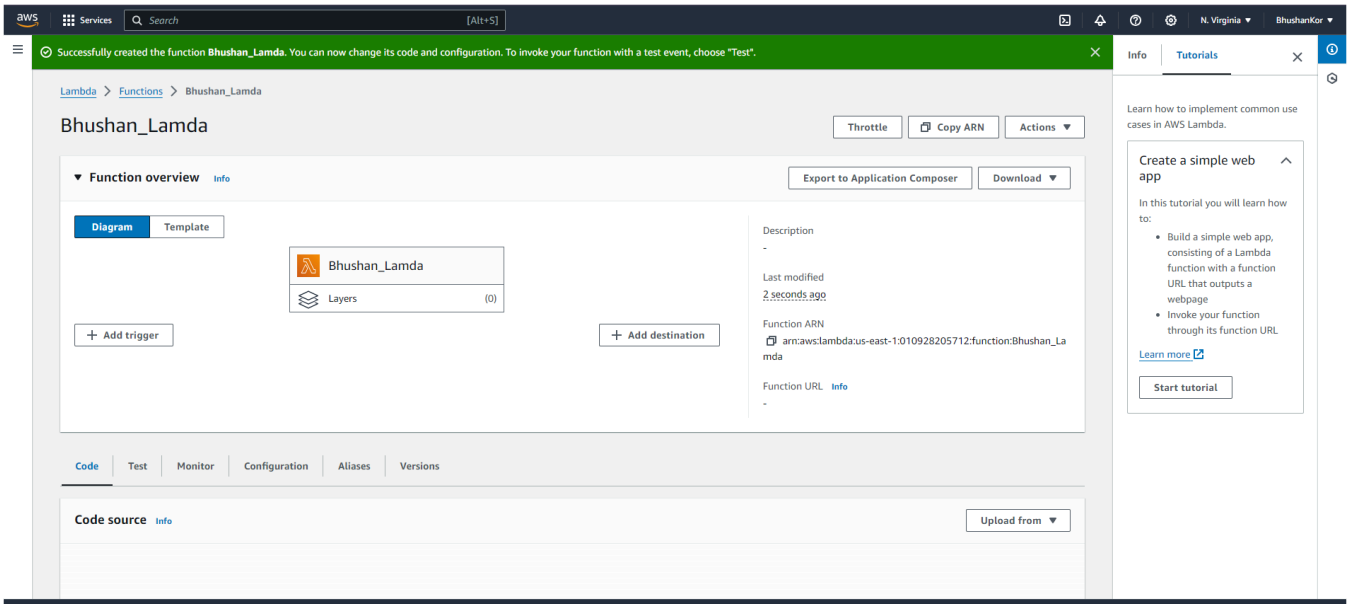
Step 4: Now Give a name to your Lambda function, Select the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby. So will select Python 3.12 , Architecture as x86, and Exceution role to Create a new role with basic Lambda permissions.

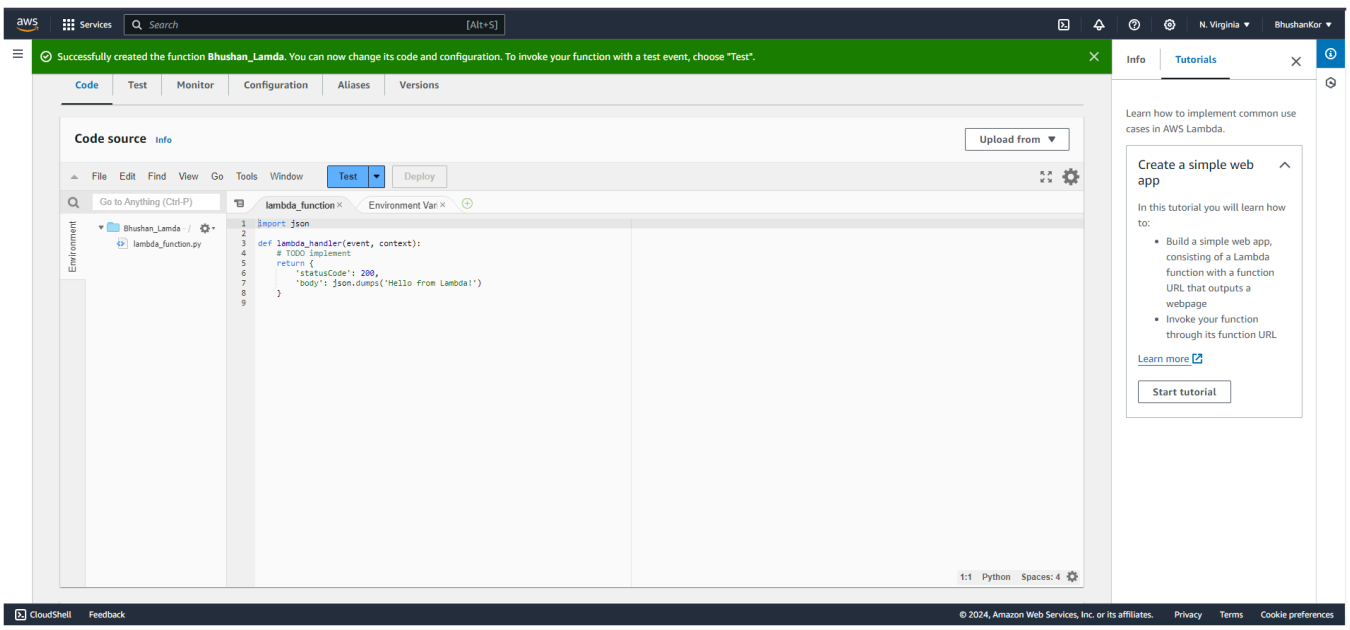


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



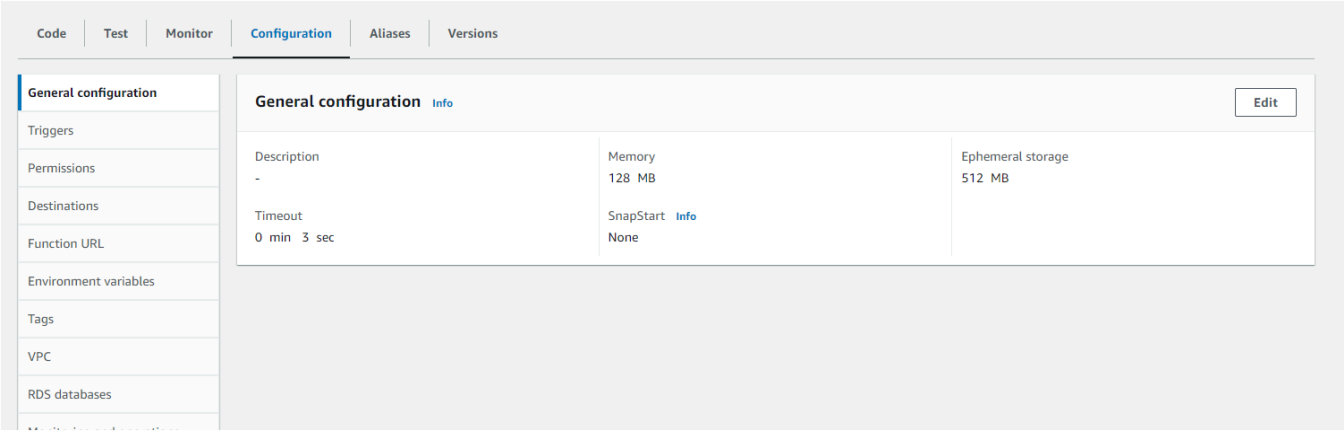




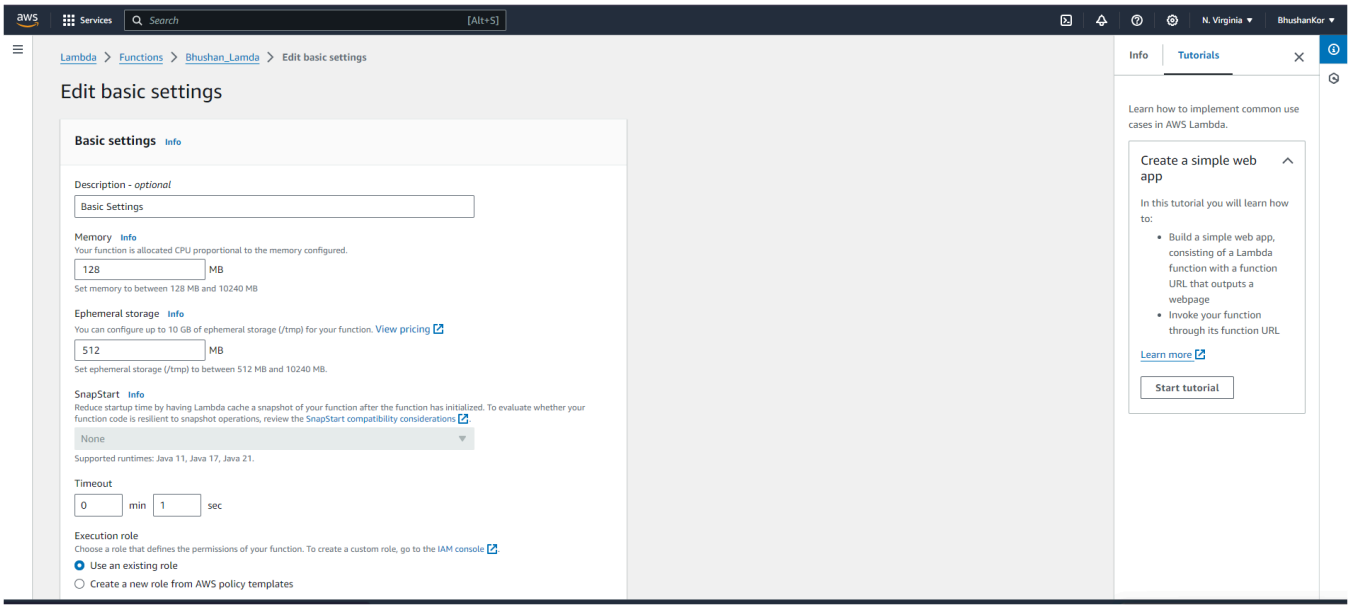
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

So See or Edit the basic settings go to configuration then click on edit general setting.



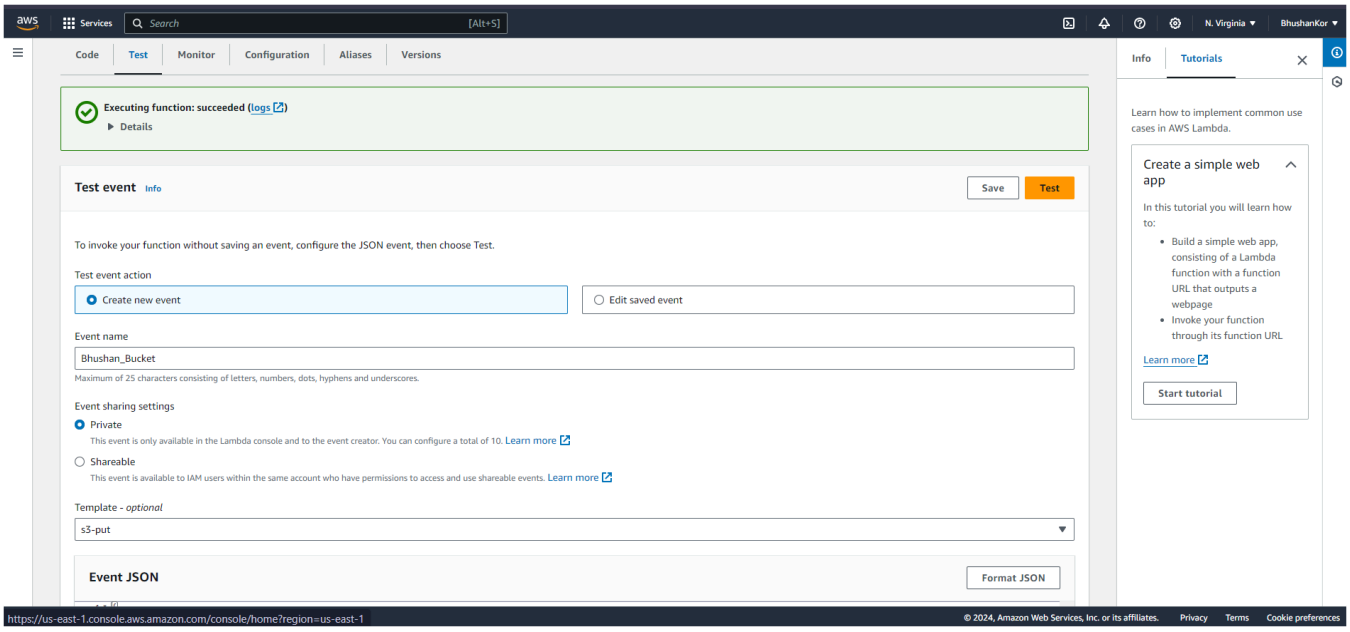
Here, you can enter a description and change Memory and Timeout. I've changed the Timeout period to 1 sec since that is sufficient for now.

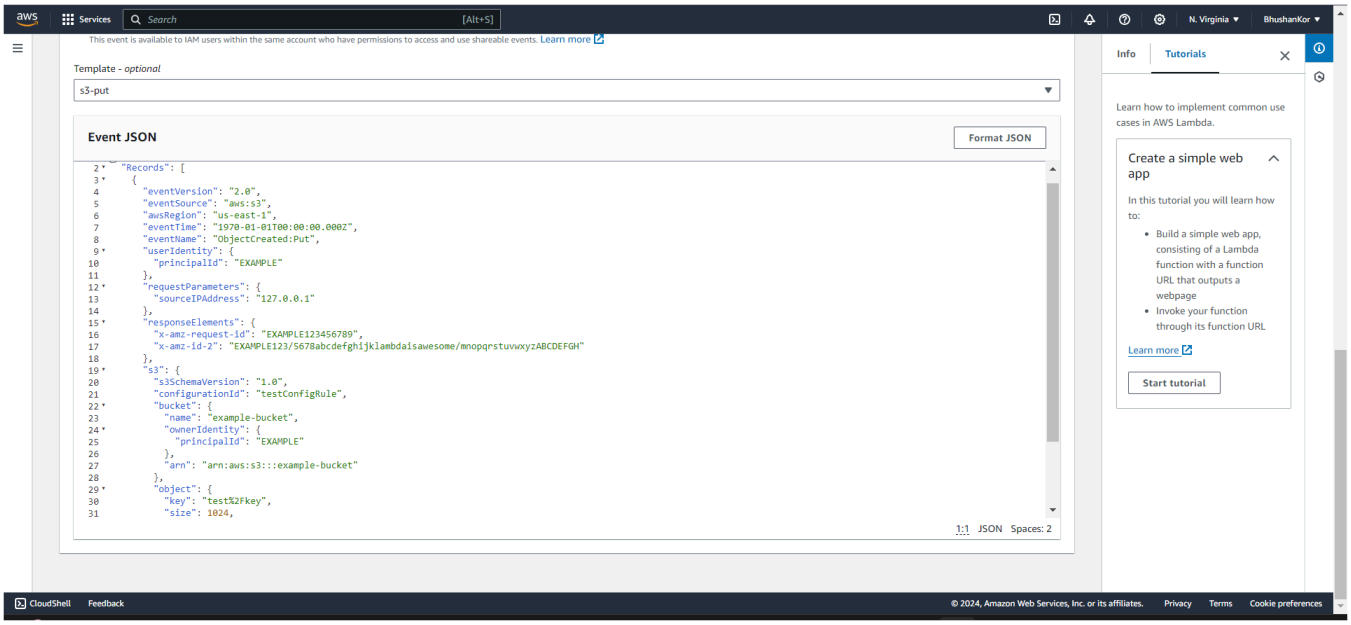


Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 5: Now Click on the Test tab then select Create a new event, give a name to the event and select Event Sharing to private, and select s3 put template.

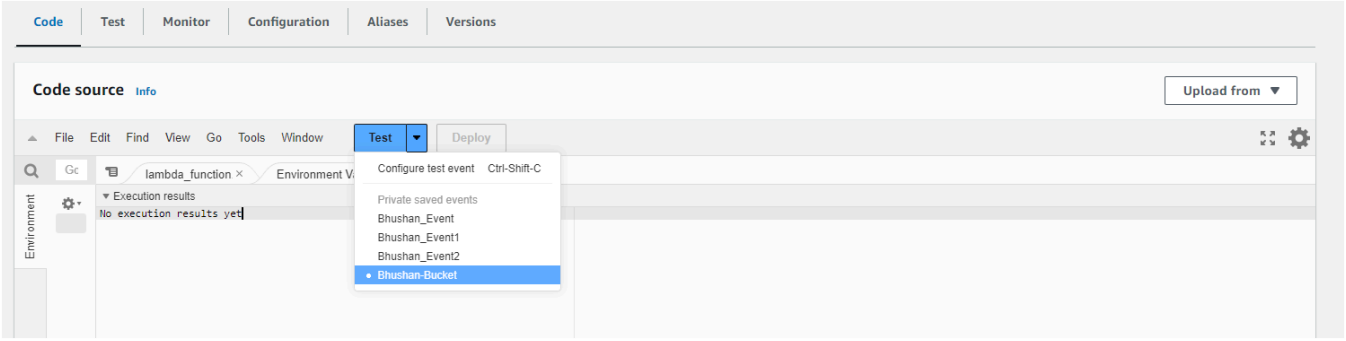




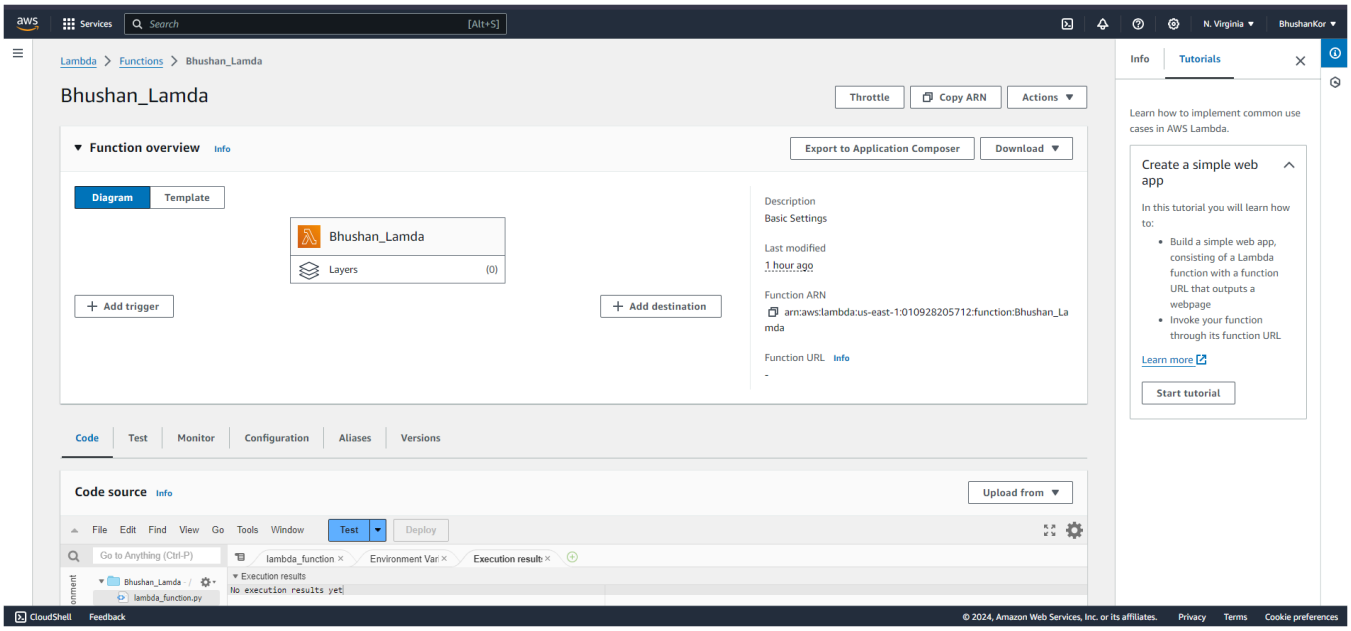
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

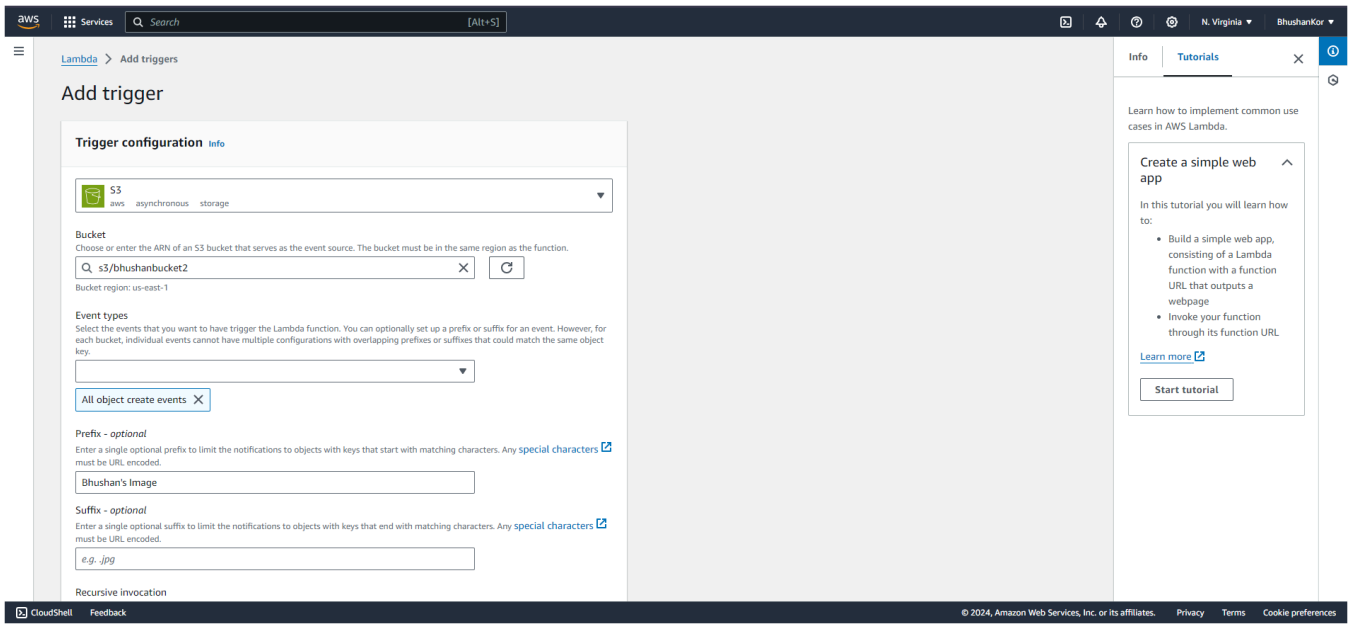
Step 6: Now In Code section select the created event from the dropdown .



Step 7: Now In the Lambda function click on add tigger.

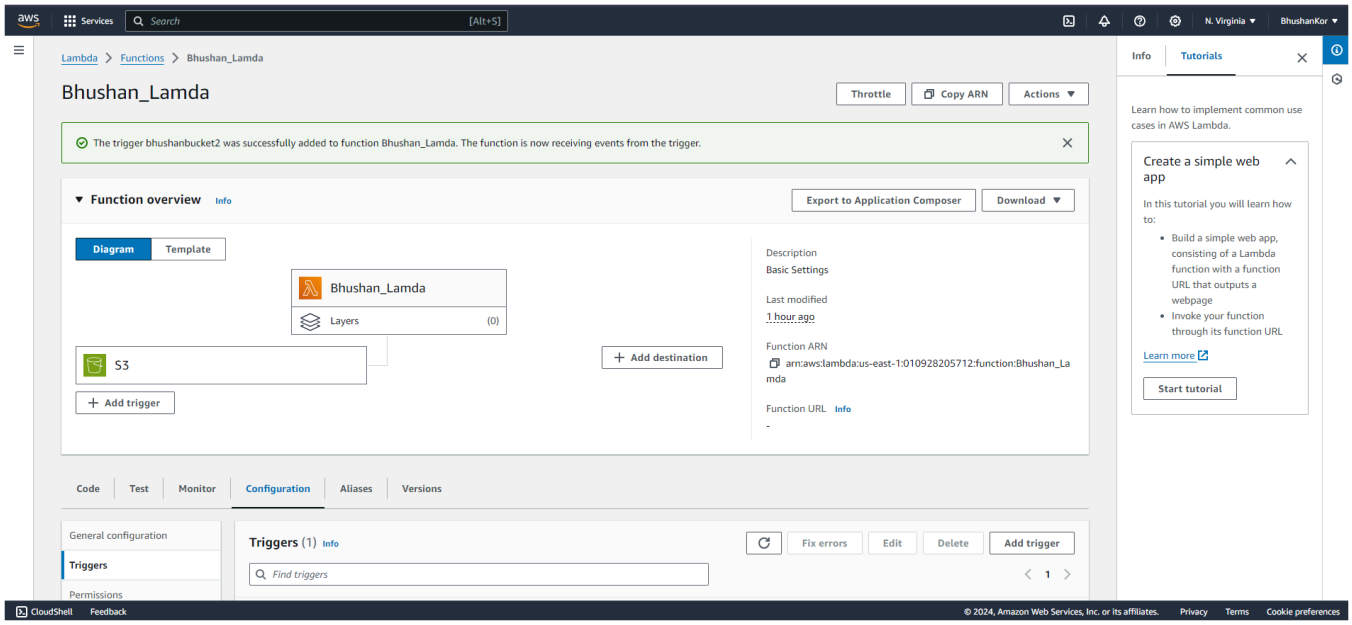


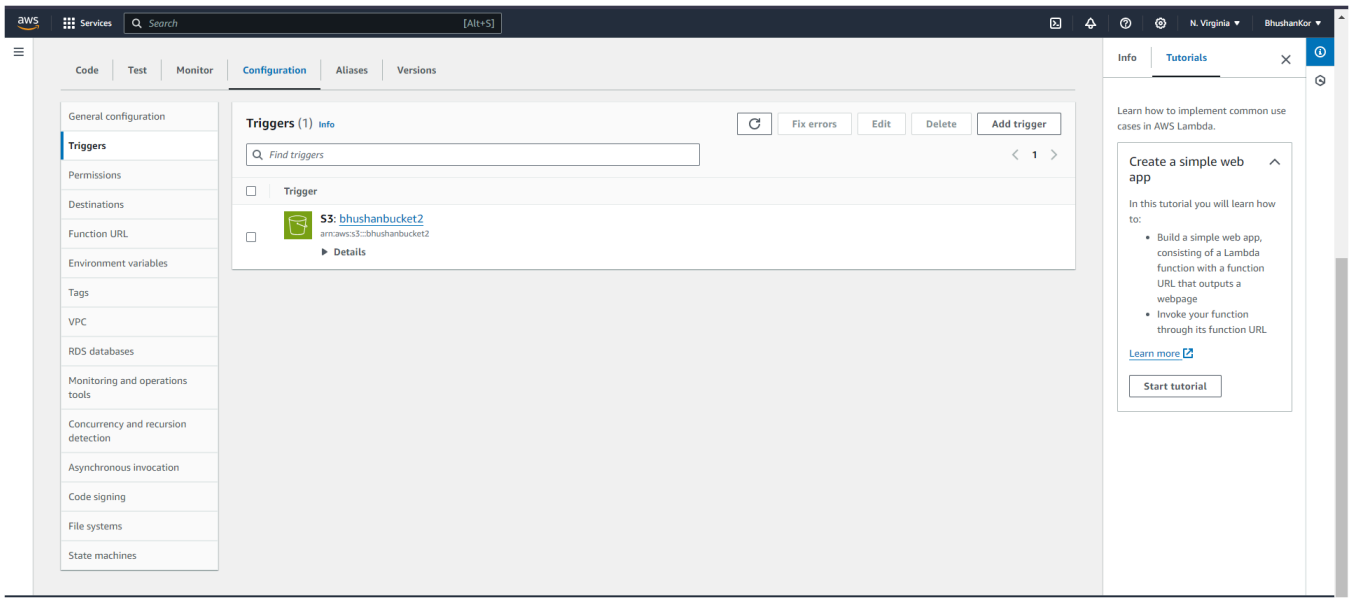
Now select the source as S3 then select the bucket name from the dropdown, keep other things to default and also you can add prefix to image.



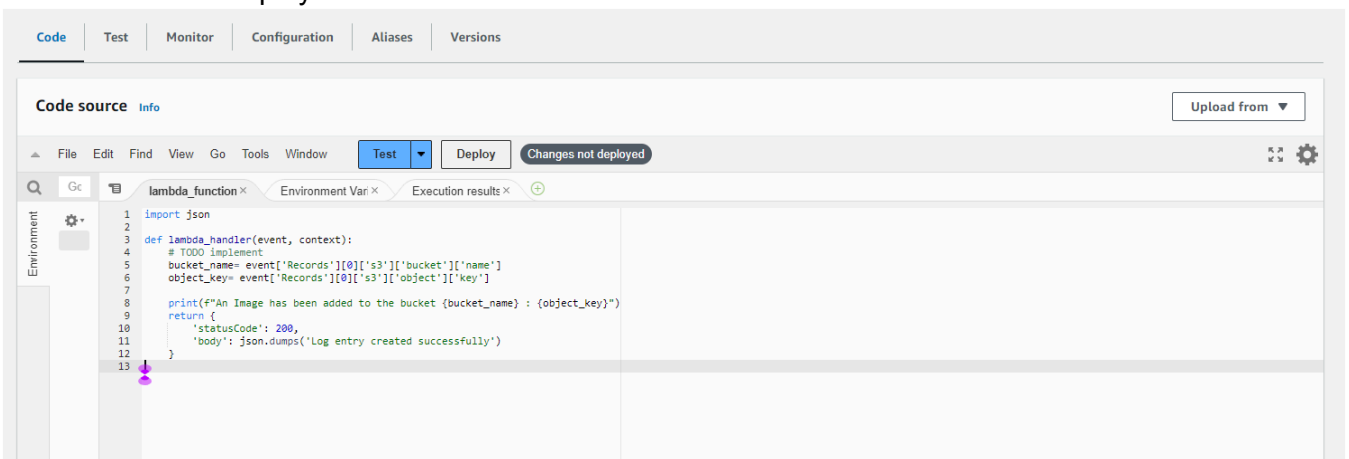
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



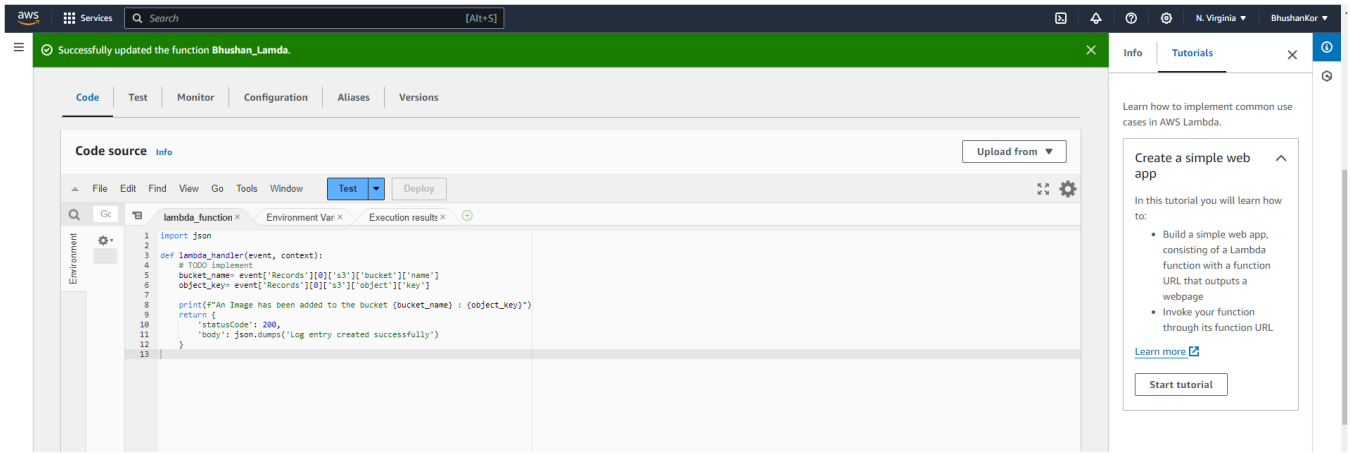


Step 8: Now Write code that logs a message like “An Image has been added” when triggered. Save the file and click on deploy.

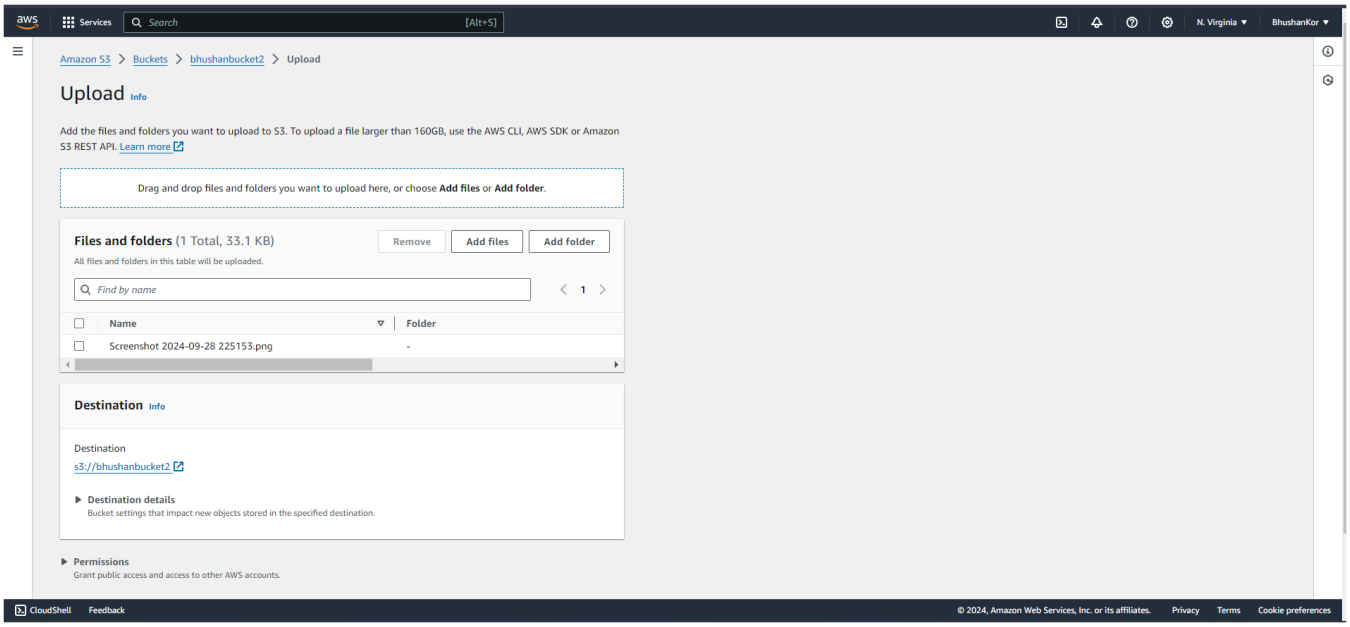


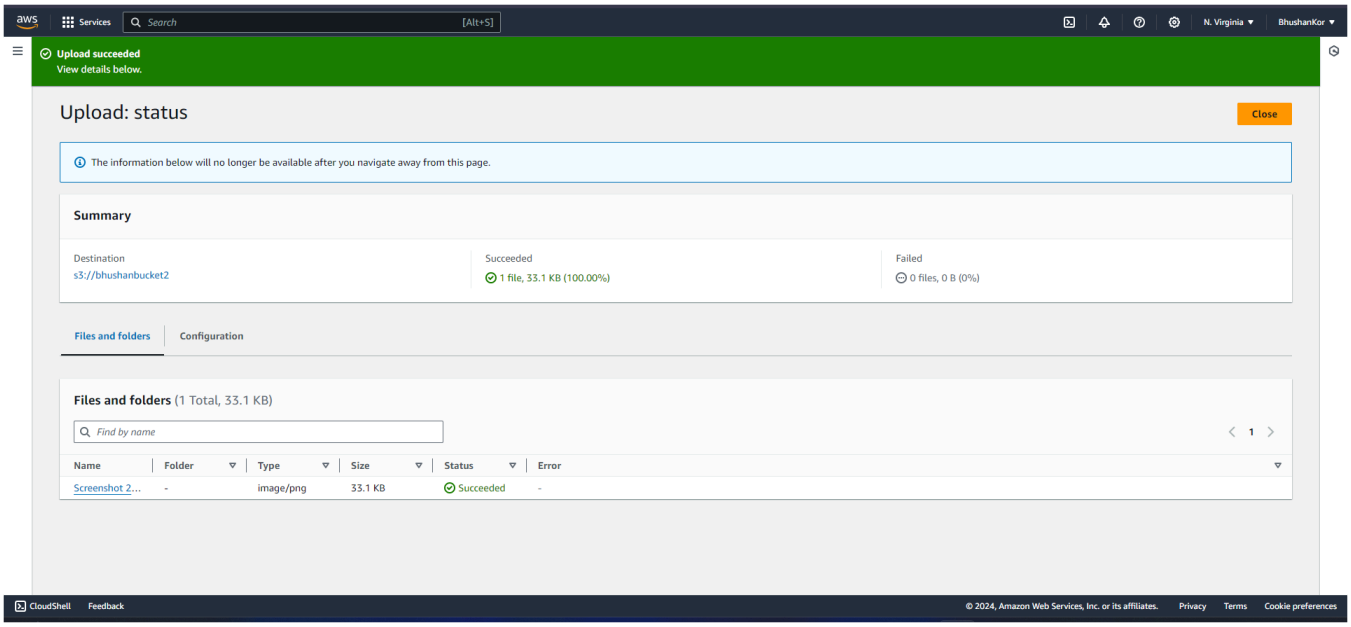
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**



Step 9: Now upload any image to the bucket.

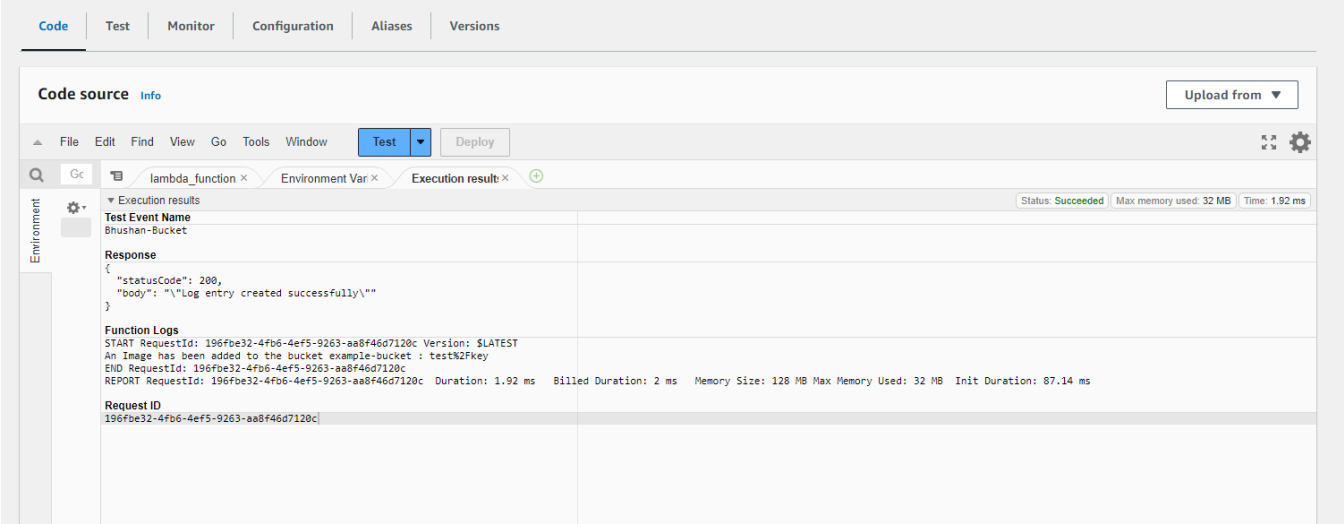




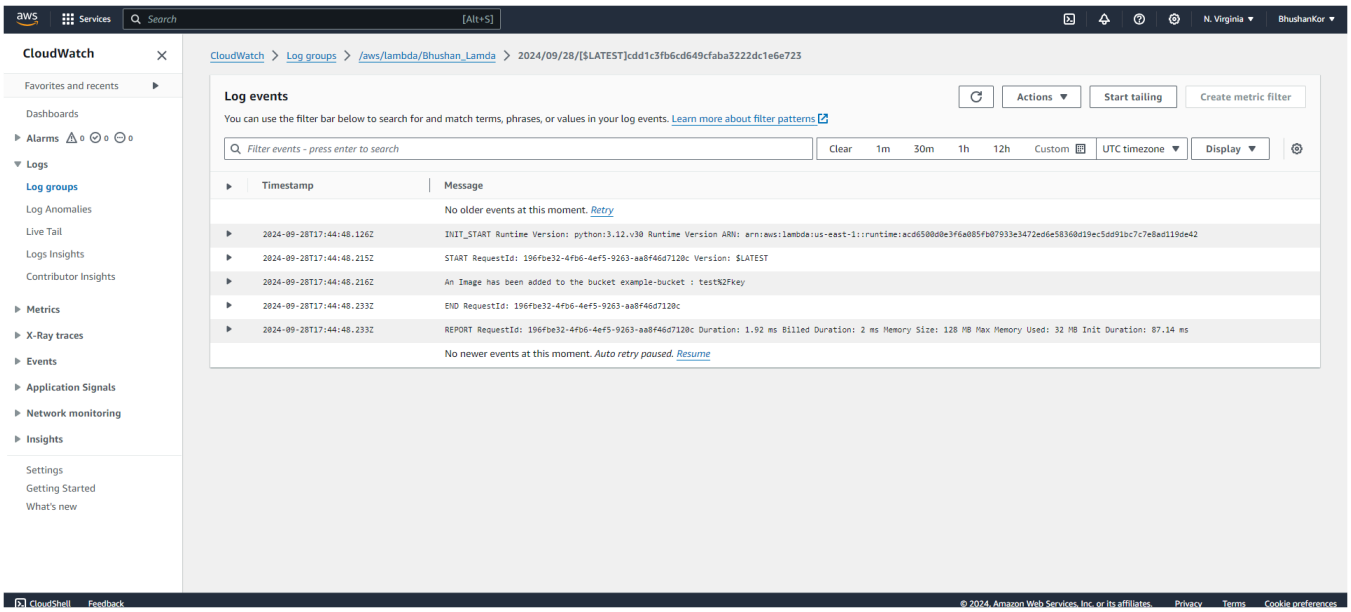
Name:Bhushan Mukund Kor **Academic Year:2024-2025**

Division: D15C **Roll No: 28**

Step 10: Now to click on test in lambda to check whether it is giving log when image is added to S3.



Step 11: Now Lets see the log on Cloud watch.To see it go to monitor section and then click on view cloudwatch logs.



Conclusion: In this experiment, we successfully created an AWS Lambda function that logs a message when an image is uploaded to an S3 bucket. It is important to note that we have to select S3-put template in event other wise code will give an error.The function was successfully triggered by S3 object uploads, validating the functionality of Lambda's event-driven architecture. This experiment demonstrated how Lambda can efficiently respond to S3 events and how to troubleshoot common issues with event structure.