

AWS CODE DEPLOY AND CODE PIPELINE

INTRODUCTION:

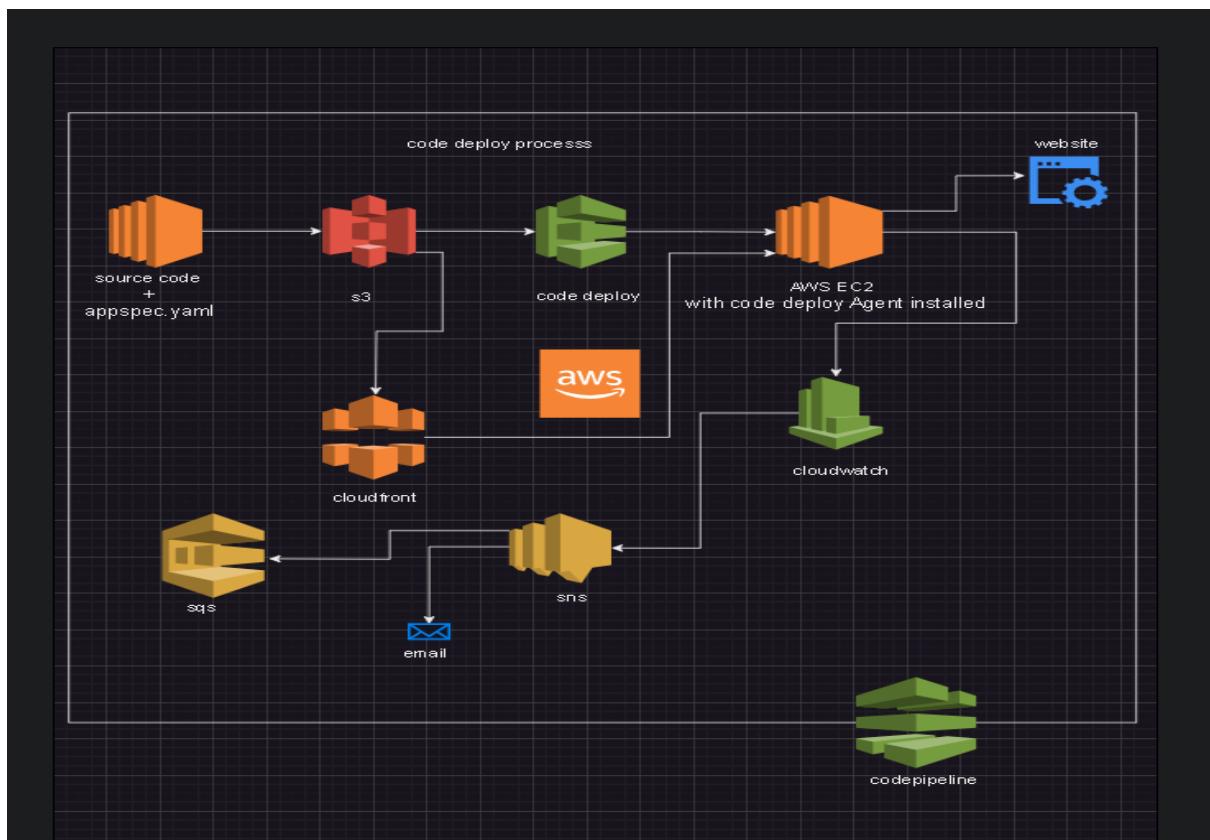
CODEDEPLOY:

AWS CodeDeploy is a service that automates application deployments to various compute services like EC2, Lambda, and on-premises servers, ensuring reliable updates and minimizing downtime.

CODEPIPELINE:

A CodePipeline automates the release process by integrating code from repositories, testing it, and deploying it to production environments, ensuring a streamlined CI/CD workflow. It enables continuous delivery of applications using AWS services.

MY PROJECT CONCEPT OUTPUT:



OBJECTIVE:

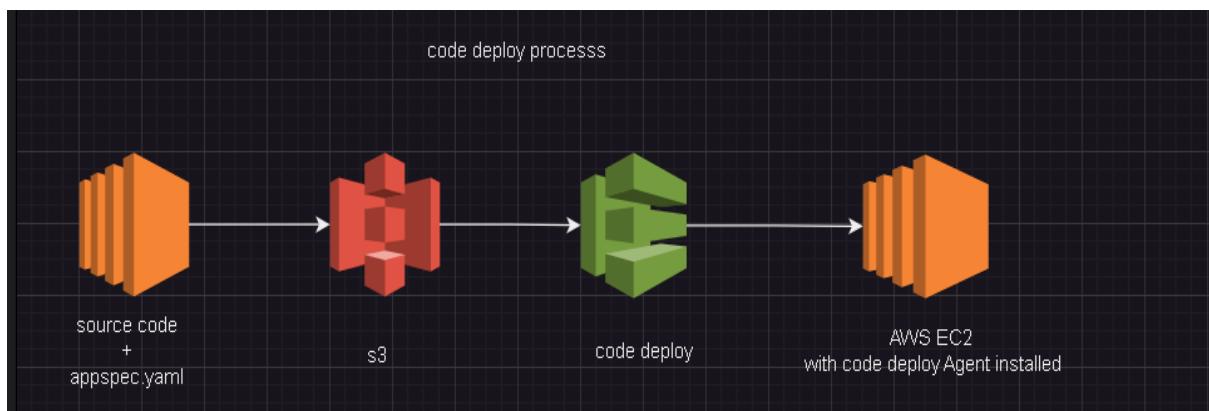
1: Streamline Continuous Deployment

"To automate and streamline the continuous deployment process using AWS CodePipeline and CodeDeploy, ensuring rapid and reliable delivery of application updates with minimal manual intervention."

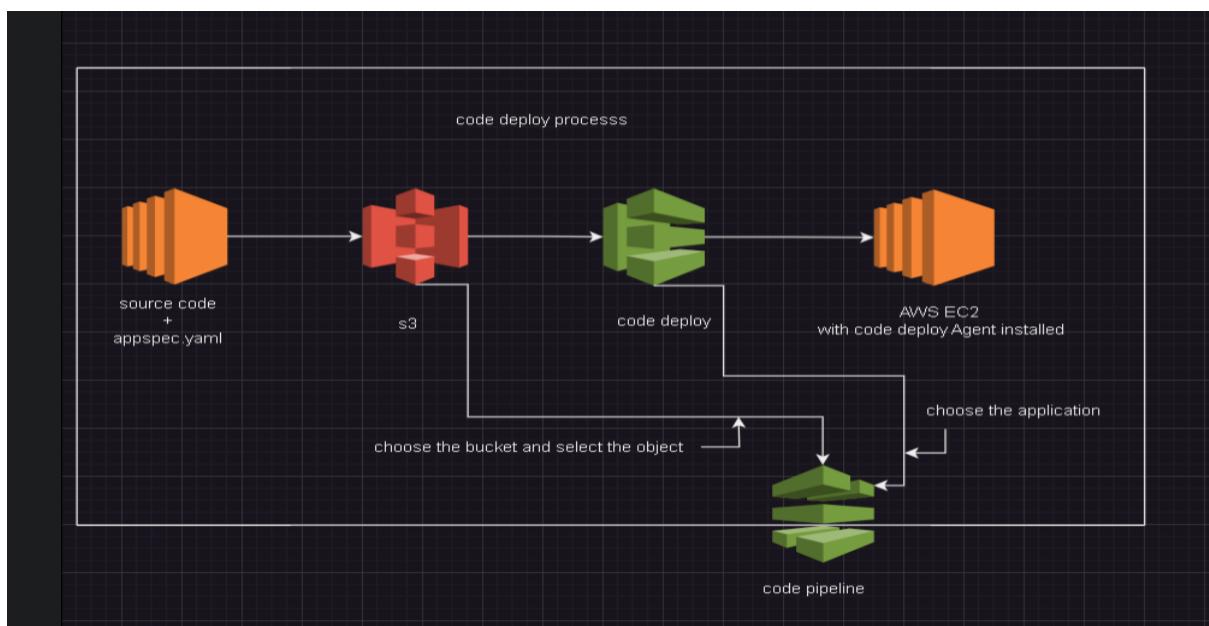
2: Enhance Deployment Efficiency

"To implement an efficient CI/CD pipeline using AWS CodePipeline and CodeDeploy, aiming to reduce deployment time, minimize errors, and ensure seamless integration and delivery of new features across multiple environments."

Code deploy:



Code pipeline:



PROJECT IMPLEMENTATION DETAILS:

CODE DEPLOY AND CODE PIPELINE PROJECT STEPS:

Setup in Brief:

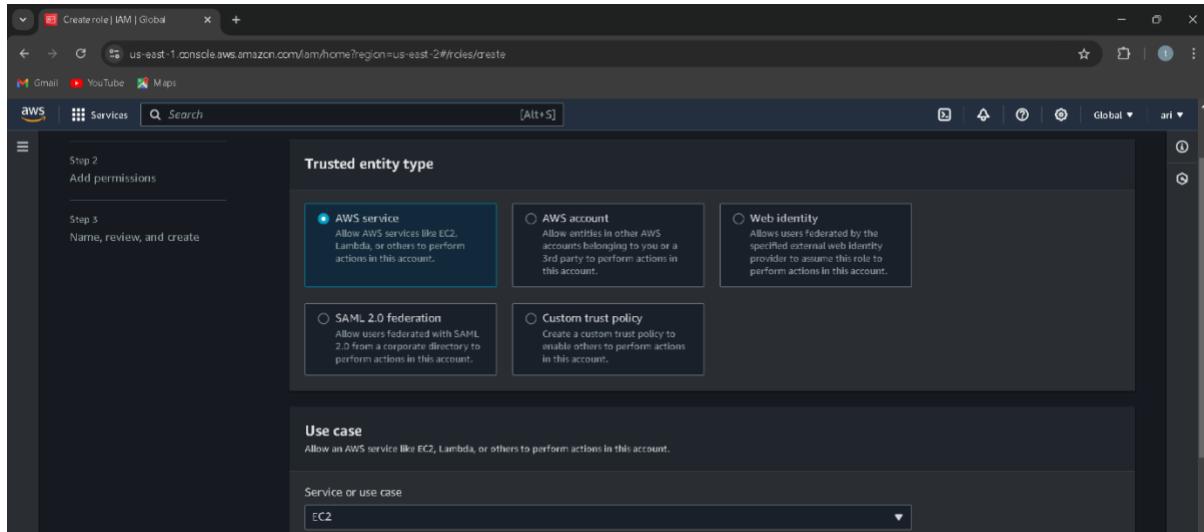
I have set up two EC2 instances of Amazon Linux 2. The first instance will serve as the web server running the CodeDeploy agent, which we will be configuring. The second EC2 instance is intended for developers to write and test code. Feel free to name the resources as you see fit for your own experiment

Project Step-By-Step Concept

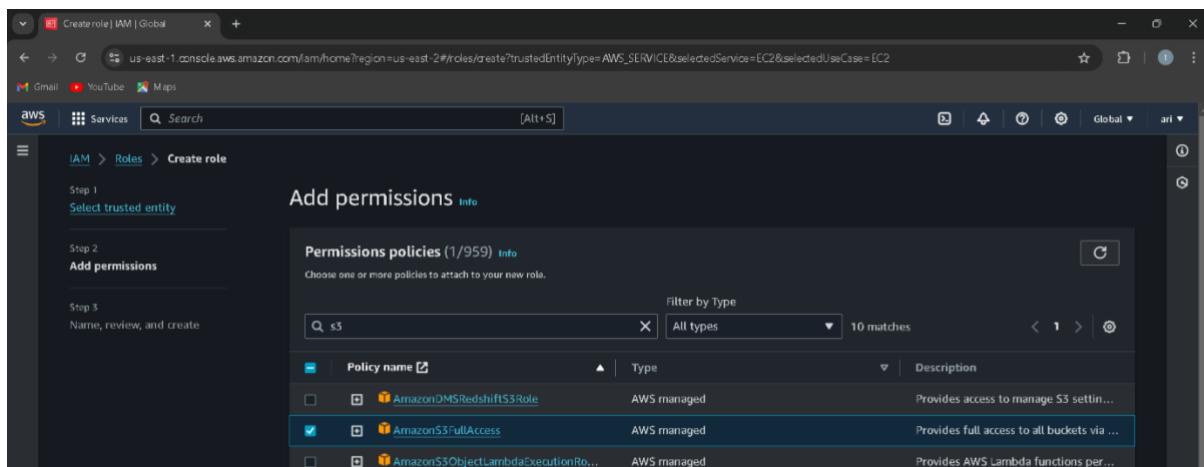
- 1) Create IAM Roles for EC2-S3-CodeDeploy access.
- 2) Create an IAM user account for the developer.
- 3) Create 2 EC2 instances – Developer Machine & Production Machine.
- 4) Connect IAM user in Developer Machine CLI.
- 5) Install the ruby in the production machine and next install the code-deploy agent.
- 6) Create a directory, yaml scripts, vi intdex.html, and script files in the developer machine.
- 7) Create an S3 Bucket to store website files.
- 8) Copy the code deploy the application and paste it onto the developer's machine.
- 9) The next one is to copy the s3 command and paste it onto the developer's machine.
- 10) The next step is to check the s3 bucket and code deploy.
- 11) Create a deployment group and the next one create a code deployment.
- 12) Successfully create the code deploy and copy the production IP address paste the IP address into your browser and visible your output
- 13) Next step is to create a code pipeline and run your output
- 14) Add additional services
- 15) Create (SQS & SNS) for Communication services.
- 16) Create a (cloud watch) for an alarm.
- 17) Create a (Cloud Front).
- 18) Create my own (portfolio website) html css.
- 19) Browse the website.

Step 1: Create IAM Roles for EC2-S3-access:

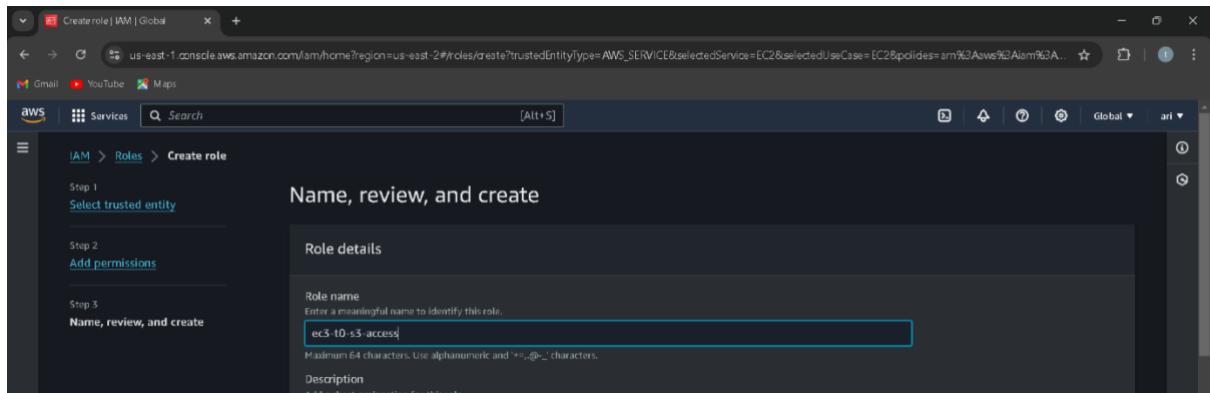
1) Select the ec2 in the use case



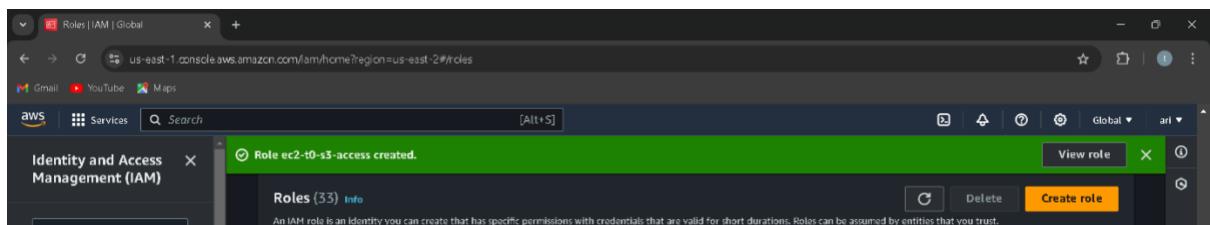
2) Select the S3 permissions.



3) Create a role name.

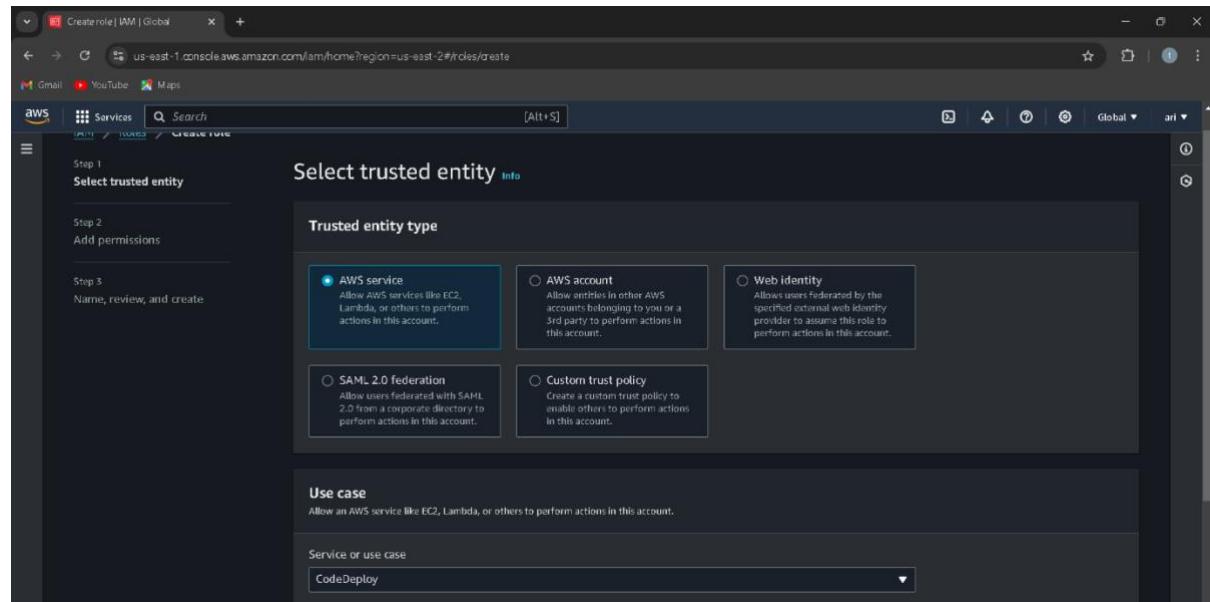


4) Successfully create the ec2-s3-access

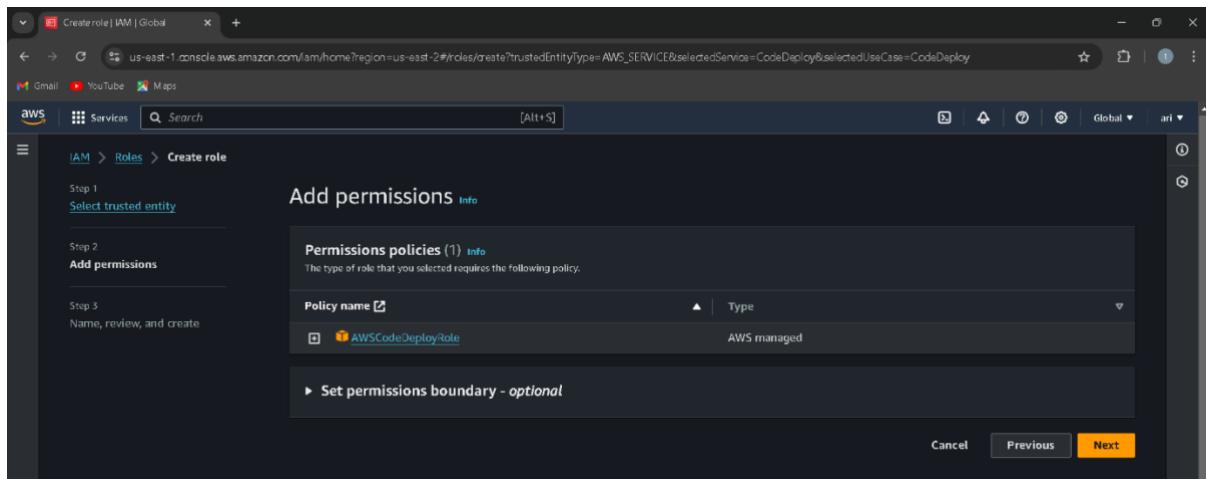


Step 2: Create a code-deploy access:

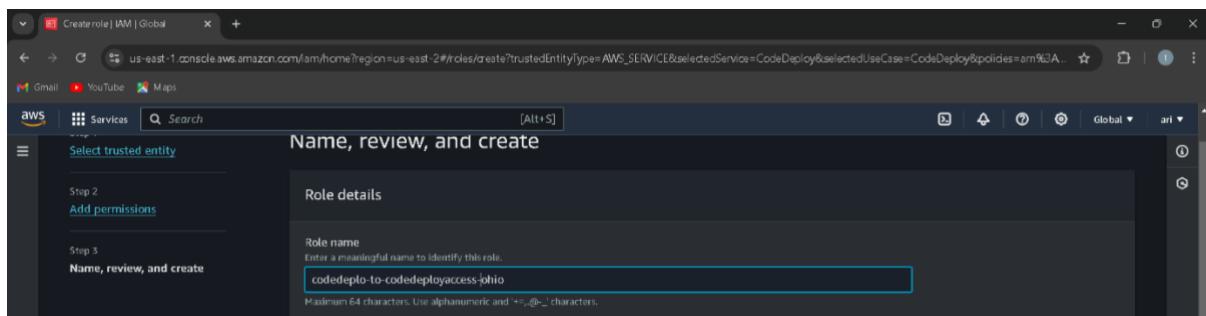
1) Select the use case in code-deploy.



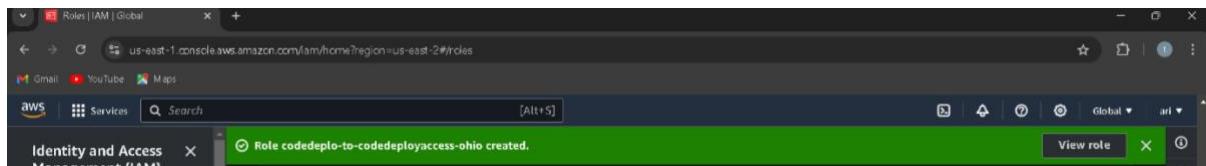
2) Add the code-deploy permission



3) Create a code-deploy access name.



4) Successfully create a code-deploy access.



Step 3: Create an IAM user:

1) Create a user name.

The screenshot shows the 'Specify user details' step of the IAM user creation wizard. The 'User name' field contains 'arisivaa'. In the 'Are you providing console access to a person?' section, the 'I want to create an IAM user' radio button is selected. The 'Console password' section shows the 'Custom password' option selected, with a password entered in the text field.

2) Next create a password.

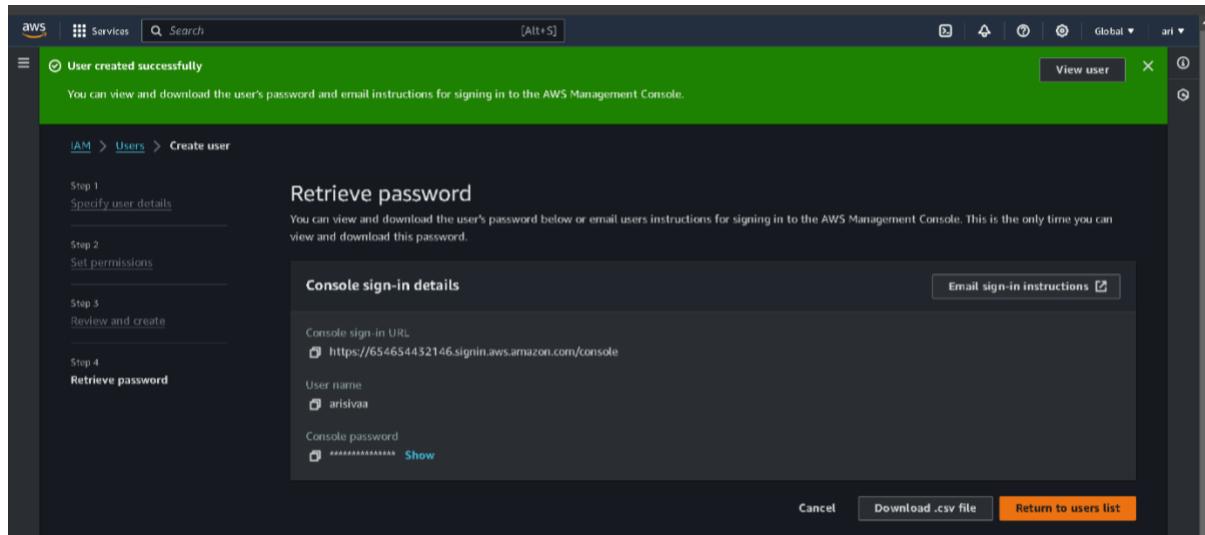
The screenshot shows the 'Specify user details' step of the IAM user creation wizard. The 'Are you providing console access to a person?' section shows the 'I want to create an IAM user' option selected. The 'Console password' section shows the 'Custom password' option selected, with a password entered in the text field.

3) Attached are the policies directly.

The screenshot shows the 'Set permissions' step of the IAM user creation wizard. The 'Permissions options' section shows the 'Attach policies directly' option selected. The 'Permissions policies' section lists the 'AdministratorAccess' policy, which is highlighted with a blue selection bar.

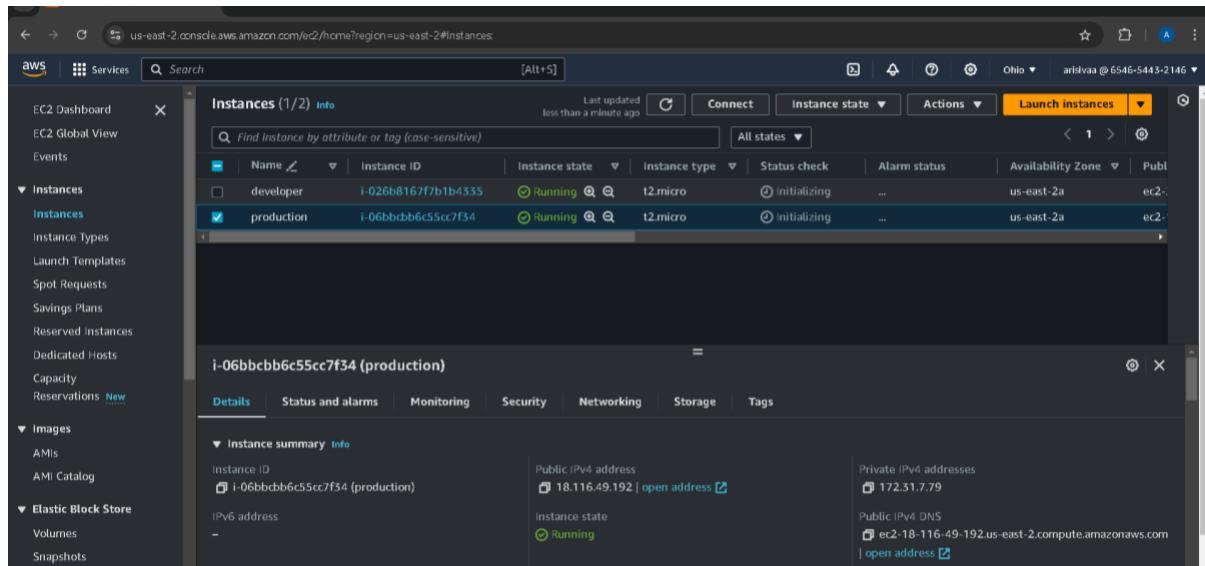
4) Successfully create an IAM user name

IAM user name : arisivaa



Step4: Create a two ec2 instances (developer&production) machine:

1) Create an Ec2 instance and the same configuration



2) The next one is to connect IAM users in the Developer machine CLI.

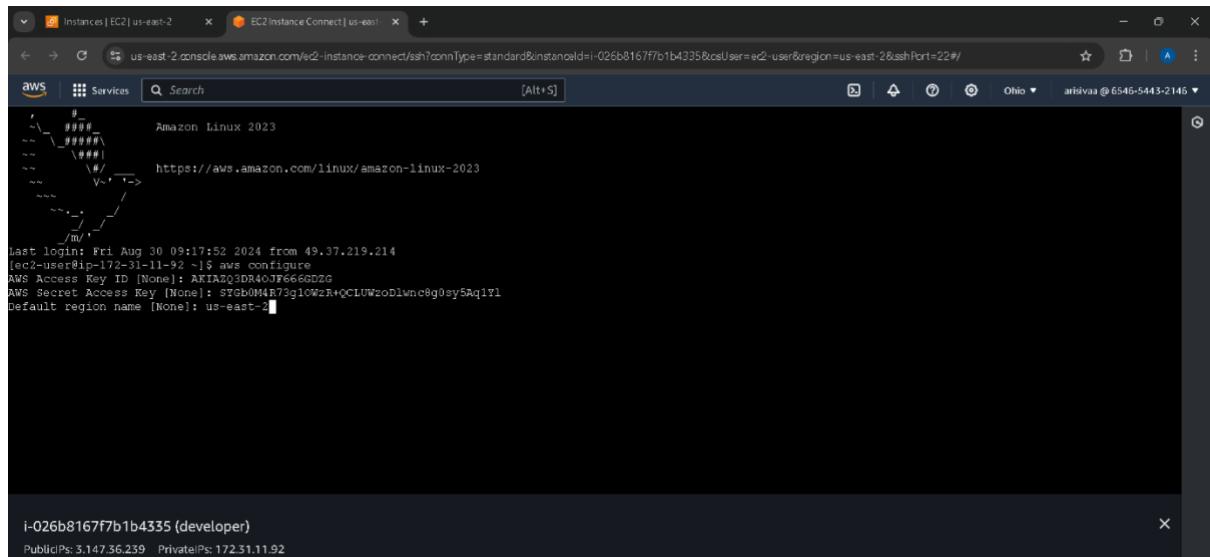
Aws configure

Acess key:

Secret key:

Region:

Default ouput:



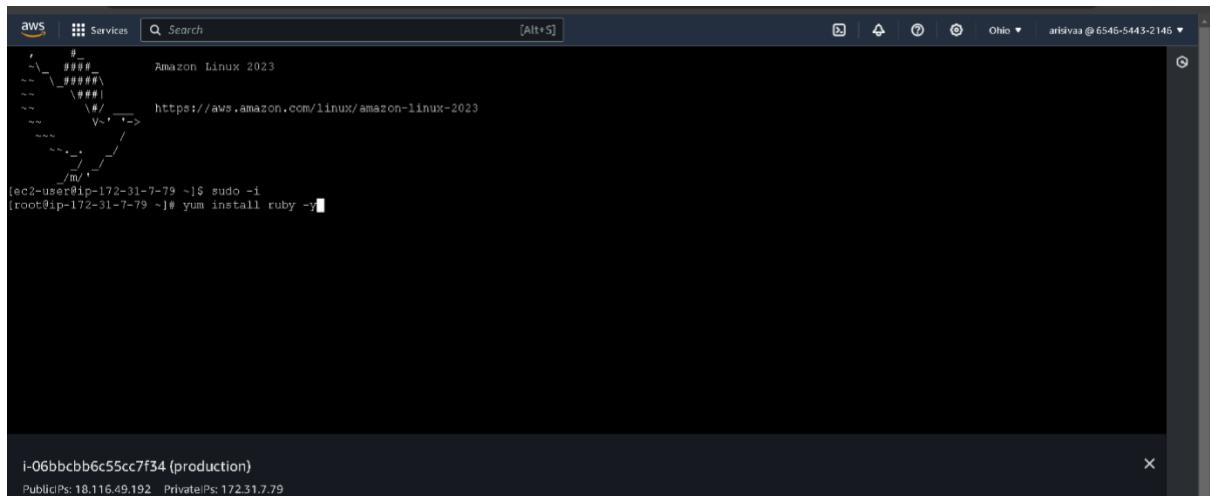
```
# Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Fri Aug 30 09:17:52 2024 from 49.37.219.214
[ec2-user@ip-172-31-11-92 ~]$ aws configure
AWS Access Key ID [None]: AXIA5Q3DR40JF666GDZG
AWS Secret Access Key [None]: STGb0M4t73g1OWzR+QCLUWzoDlwnc0g0sy5Aq1Yl
Default region name [None]: us-east-1

i-026b8167f7b1b4335 (developer)
PublicIPs: 3.147.36.259 PrivateIPs: 172.31.11.92
```

3) Install the ruby in the production machine and next install the code-deploy age

Ruby install command: **Yum install ruby -y**



```
# Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-172-31-7-79 ~]$ sudo -i
[root@ip-172-31-7-79 ~]# yum install ruby -y

i-06bbccbb6c55cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.31.7.79
```

Ruby installation complete.

```

Instances | EC2 | us-east-2 EC2 Instance Connect | us-east-2 EC2 Instance Connect | us-east-2
us-east-2.console.aws.amazon.com/ec2-instance-connect/sh?region=us-east-2&connType=standard&instanceId=i-06bbcb6c5cc7f34&csUser=ec2-user&sshPort=22#
aws Services Search [Alt+S]
Installing : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 8/10
Installing : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 9/10
Installing : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Verifying : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 1/10
Verifying : ruby3.2-default-gems-3.2.2-180.amzn2023.0.3.noarch 2/10
Verifying : ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64 3/10
Verifying : ruby3.2-rubygem-bigdecimal-3.1.3-180.amzn2023.0.3.x86_64 4/10
Verifying : ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 5/10
Verifying : ruby3.2-rubygem-io-console-0.6.0-180.amzn2023.0.3.x86_64 6/10
Verifying : ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64 7/10
Verifying : ruby3.2-rubygem-psych-5.0.1-180.amzn2023.0.3.x86_64 8/10
Verifying : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 9/10
Verifying : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10

Installed:
ruby3.2-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch
ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch

Complete!
[root@ip-172-31-7-79 ~]# 

I-06bbcb6c5cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.51.7.9

```

```

Instances | EC2 | us-east-2 EC2 Instance Connect | us-east-2 EC2 Instance Connect | us-east-2
us-east-2.console.aws.amazon.com/ec2-instance-connect/sh?region=us-east-2&connType=standard&instanceId=i-06bbcb6c5cc7f34&csUser=ec2-user&sshPort=22#
aws Services Search [Alt+S]
Installing : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 8/10
Installing : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 9/10
Installing : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 1/10
Verifying : ruby3.2-3.2.2-180.amzn2023.0.3.noarch 2/10
Verifying : ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64 3/10
Verifying : ruby3.2-rubygem-bigdecimal-3.1.3-180.amzn2023.0.3.x86_64 4/10
Verifying : ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 5/10
Verifying : ruby3.2-rubygem-io-console-0.6.0-180.amzn2023.0.3.x86_64 6/10
Verifying : ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64 7/10
Verifying : ruby3.2-rubygem-psych-5.0.1-180.amzn2023.0.3.x86_64 8/10
Verifying : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 9/10
Verifying : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10

Installed:
ruby3.2-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch
ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch

Complete!
[root@ip-172-31-7-79 ~]# 

I-06bbcb6c5cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.51.7.9

```

```

Instances | EC2 | us-east-2 EC2 Instance Connect | us-east-2 EC2 Instance Connect | us-east-2
us-east-2.console.aws.amazon.com/ec2-instance-connect/sh?region=us-east-2&connType=standard&instanceId=i-06bbcb6c5cc7f34&csUser=ec2-user&sshPort=22#
aws Services Search [Alt+S]
Installing : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 8/10
Installing : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 9/10
Installing : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 1/10
Verifying : ruby3.2-3.2.2-180.amzn2023.0.3.noarch 2/10
Verifying : ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64 3/10
Verifying : ruby3.2-rubygem-bigdecimal-3.1.3-180.amzn2023.0.3.x86_64 4/10
Verifying : ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 5/10
Verifying : ruby3.2-rubygem-io-console-0.6.0-180.amzn2023.0.3.x86_64 6/10
Verifying : ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64 7/10
Verifying : ruby3.2-rubygem-psych-5.0.1-180.amzn2023.0.3.x86_64 8/10
Verifying : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 9/10
Verifying : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10

Installed:
ruby3.2-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch
ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch

Complete!
[root@ip-172-31-7-79 ~]# 

I-06bbcb6c5cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.51.7.9

```

```

aws Services Search [Alt+S] Instances | EC2 | us-east-2 EC2 Instance Connect | us-east-2 EC2 Instance Connect | us-east-2
us-east-2.console.aws.amazon.com/ec2-instance-connect/sh?region=us-east-2&connType=standard&instanceId=i-06bbcb6c5cc7f34&caUser=ec2-user&sshPort=22#
arivaa @ 6546-5443-2146

Installing : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 8/10
Installing : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 9/10
Installing : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Verifying : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 1/10
Verifying : ruby3.2-default-gems-3.2.2-180.amzn2023.0.3.noarch 2/10
Verifying : ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64 3/10
Verifying : ruby3.2-rubygem-bigdecimal-3.1.3-180.amzn2023.0.3.x86_64 4/10
Verifying : ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 5/10
Verifying : ruby3.2-rubygem-io-console-0.6.0-180.amzn2023.0.3.x86_64 6/10
Verifying : ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64 7/10
Verifying : ruby3.2-rubygem-psych-5.0.1-180.amzn2023.0.3.x86_64 8/10
Verifying : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 9/10
Verifying : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10

Installed:
ruby3.2-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch
ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch

Completed!
[root@ip-172-31-7-79 ~]# wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
l-06bbcb6c5cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.31.7.99

```

4) Already wget a command package installed on my server

(COPY THE CODE AGENT COMMENT AND PASTE)

1) Copy the code deploy agent and paste it to your server.

```

aws Services Search [Alt+S] Instances | EC2 | us-east-2 EC2 Instance Connect | us-east-2 EC2 Instance Connect | us-east-2 Install the CodeDeploy agent
us-east-2.console.aws.amazon.com/ec2-instance-connect/sh?region=us-east-2&connType=standard&instanceId=i-06bbcb6c5cc7f34&caUser=ec2-user&sshPort=22#
arivaa @ 6546-5443-2146

Installing : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 8/10
Installing : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 9/10
Installing : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Running scriptlet: ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10
Running scriptlet: ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 10/10
Verifying : ruby3.2-3.2.2-180.amzn2023.0.3.x86_64 1/10
Verifying : ruby3.2-default-gems-3.2.2-180.amzn2023.0.3.noarch 2/10
Verifying : ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64 3/10
Verifying : ruby3.2-rubygem-bigdecimal-3.1.3-180.amzn2023.0.3.x86_64 4/10
Verifying : ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch 5/10
Verifying : ruby3.2-rubygem-io-console-0.6.0-180.amzn2023.0.3.x86_64 6/10
Verifying : ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64 7/10
Verifying : ruby3.2-rubygem-psych-5.0.1-180.amzn2023.0.3.x86_64 8/10
Verifying : ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch 9/10
Verifying : ruby3.2-rubygems-3.4.10-180.amzn2023.0.3.noarch 10/10

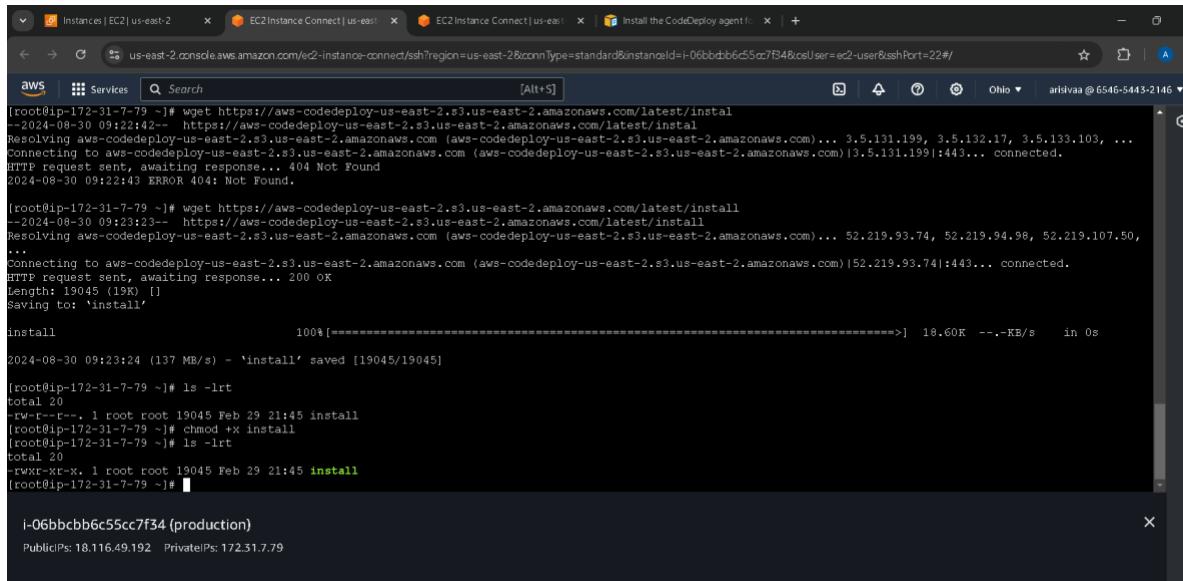
Installed:
ruby3.2-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-libs-3.2.2-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-bundler-2.4.10-180.amzn2023.0.3.noarch
ruby3.2-rubygem-json-2.6.3-180.amzn2023.0.3.x86_64
ruby3.2-rubygem-rdoc-6.5.0-180.amzn2023.0.3.noarch

Completed!
[root@ip-172-31-7-79 ~]# wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
l-06bbcb6c5cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.31.7.99

```

Wget aws deploy create-application --application-name sampleapp

2) Successfully created a code deploy-agent



```
[root@ip-172-31-7-79 ~]# wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
--2024-08-30 09:22:42-- https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
Resolving aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com)... 3.5.131.199, 3.5.132.17, 3.5.133.103, ...
Connecting to aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com) [3.5.131.199]:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2024-08-30 09:22:43 ERROR 404: Not Found.

[root@ip-172-31-7-79 ~]# wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
--2024-08-30 09:23:23-- https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
Resolving aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com)... 52.219.93.74, 52.219.94.98, 52.219.107.50,
...
Connecting to aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com) [52.219.93.74]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 19045 (19KB) []
saving to: 'install'

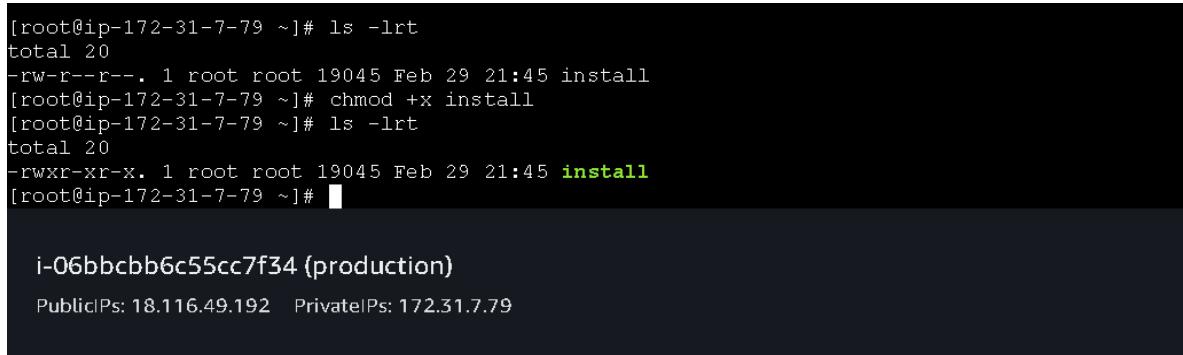
install          100%[=====] 18.60K --.-KB/s   in 0s

2024-08-30 09:23:24 (137 MB/s) - 'install' saved [19045/19045]

[root@ip-172-31-7-79 ~]# ls -lrt
total 20
-rw-r--r--. 1 root root 19045 Feb 29 21:45 install
[root@ip-172-31-7-79 ~]# chmod +x install
[root@ip-172-31-7-79 ~]# ls -lrt
total 20
-rwxr-xr-x. 1 root root 19045 Feb 29 21:45 install
[root@ip-172-31-7-79 ~]# 
```

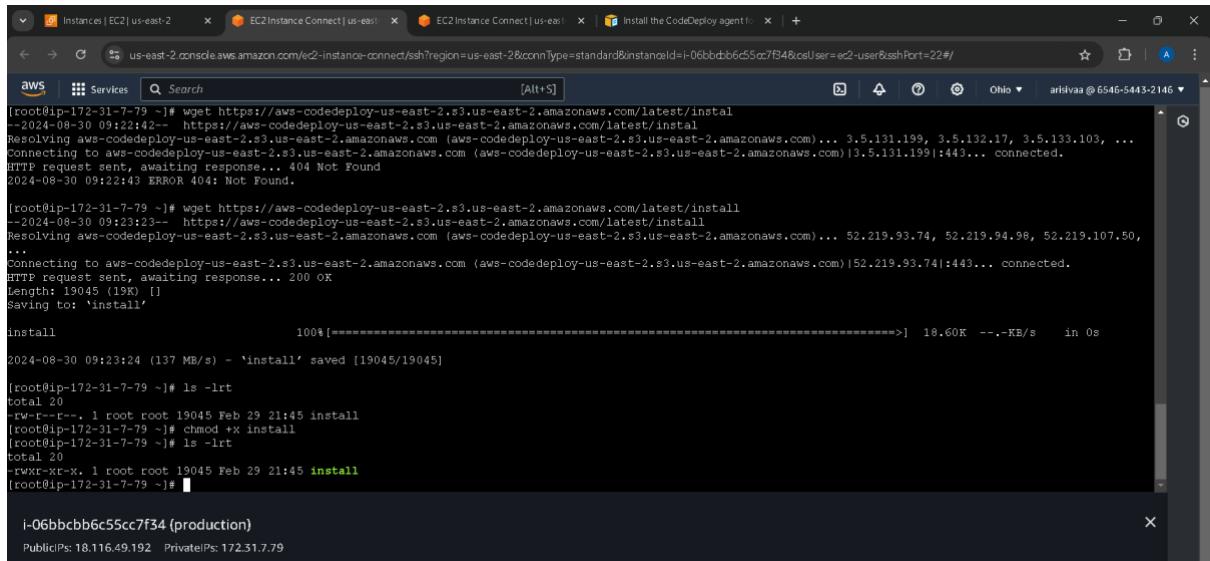
i-06bbcb6c55cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.31.7.79

3) Change the file mode using this command (**chmod +x install**)



```
[root@ip-172-31-7-79 ~]# ls -lrt
total 20
-rw-r--r--. 1 root root 19045 Feb 29 21:45 install
[root@ip-172-31-7-79 ~]# chmod +x install
[root@ip-172-31-7-79 ~]# ls -lrt
total 20
-rwxr-xr-x. 1 root root 19045 Feb 29 21:45 install
[root@ip-172-31-7-79 ~]# 
```

i-06bbcb6c55cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.31.7.79



```
[root@ip-172-31-7-79 ~]# wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
--2024-08-30 09:22:42-- https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
Resolving aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com)... 3.5.131.199, 3.5.132.17, 3.5.133.103, ...
Connecting to aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com) [3.5.131.199]:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2024-08-30 09:22:43 ERROR 404: Not Found.

[root@ip-172-31-7-79 ~]# wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
--2024-08-30 09:23:23-- https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
Resolving aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com)... 52.219.93.74, 52.219.94.98, 52.219.107.50,
...
Connecting to aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com (aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com) [52.219.93.74]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 19045 (19KB) []
saving to: 'install'

install          100%[=====] 18.60K --.-KB/s   in 0s

2024-08-30 09:23:24 (137 MB/s) - 'install' saved [19045/19045]

[root@ip-172-31-7-79 ~]# ls -lrt
total 20
-rw-r--r--. 1 root root 19045 Feb 29 21:45 install
[root@ip-172-31-7-79 ~]# chmod +x install
[root@ip-172-31-7-79 ~]# ls -lrt
total 20
-rwxr-xr-x. 1 root root 19045 Feb 29 21:45 install
[root@ip-172-31-7-79 ~]# 
```

i-06bbcb6c55cc7f34 (production)
PublicIPs: 18.116.49.192 PrivateIPs: 172.31.7.79


```
# Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Fri Aug 30 09:17:52 2024 from 49.37.219.214
[ec2-user@ip-172-31-11-92 ~]$ aws configure
AWS Access Key ID [None]: AKIAZQ3DR40JF66GDZG
AWS Secret Access Key [None]: STGb0M4R73g10WzR+QCLUWzoD1wnc8gOsy5Aq1Yl
Default Region Name [None]: us-east-2
Default output format [None]: json
[ec2-user@ip-172-31-11-92 ~]$ ls s3 ls
2024-08-30 06:09:47 bucketuseast-2
[ec2-user@ip-172-31-11-92 ~]$ mkdir deploy_dir
[ec2-user@ip-172-31-11-92 ~]$ cd deploy_dir
[ec2-user@ip-172-31-11-92 deploy_dir]$ mkdir sampleapp
[ec2-user@ip-172-31-11-92 deploy_dir]$ cd sampleapp
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
[ec2-user@ip-172-31-11-92 sampleapp]$ vi appspec.yaml
[ec2-user@ip-172-31-11-92 sampleapp]$ mkdir scripts
[ec2-user@ip-172-31-11-92 sampleapp]$ cd scripts
[ec2-user@ip-172-31-11-92 scripts]$
```

i-026b8167f7b1b4335 (developer)
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92

2) vi index.html

```
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
```

Enter the content below and save by typing :wq! and pressing Enter

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Arisiva Sankar's Portfolio</title>
    <style>
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
}

header {
    background-color: black;
    color: #fff;
    padding: 1em 0;
    text-align: center;
}

header h1 {
    margin: 0;
}

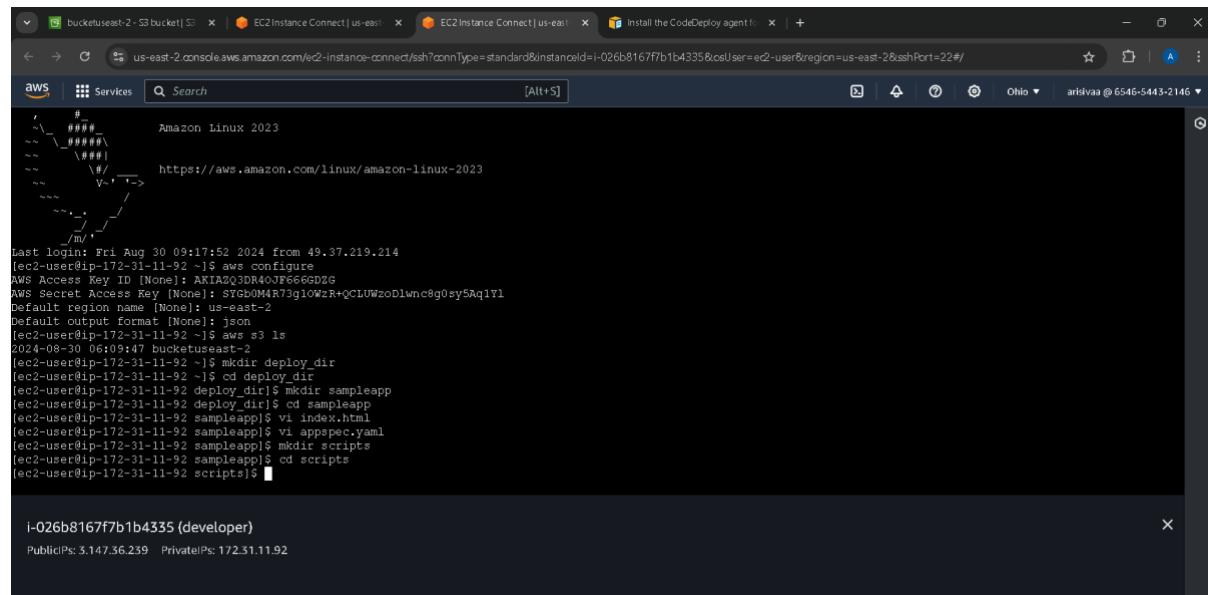
nav ul {
```

:wq! Then Enter

Step 6: Create appspec.yaml file: This file helps in automatically deploying the source code to The webserver.

1) Vi appspec.yaml

```
e2-user@ip-172-31-11-92 sampleapp]$ vi appspec.yaml
```



```
#!/bin/sh
# Amazon Linux 2023
## 
## https://aws.amazon.com/linux/amazon-linux-2023
## 

Last login: Fri Aug 30 09:17:52 2024 from 49.37.219.214
[ec2-user@ip-172-31-11-92 ~]$ aws configure
AWS Access Key ID [None]: AKIAZQ3DR40JF66GDZG
AWS Secret Access Key [None]: STGb0M4R73g10WzR+QCLUWzoDlwnC8g0sy5Aq1Y1
default region name [None]: us-east-2
default output format [None]: json
[ec2-user@ip-172-31-11-92 ~]$ aws s3 ls
2024-08-30 06:09:47 bucketuseast-2
[ec2-user@ip-172-31-11-92 ~]$ mkdir deploy_dir
[ec2-user@ip-172-31-11-92 deploy_dir]$ cd sampleapp
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
[ec2-user@ip-172-31-11-92 sampleapp]$ vi appspec.yaml
[ec2-user@ip-172-31-11-92 sampleapp]$ mkdir scripts
[ec2-user@ip-172-31-11-92 sampleapp]$ cd scripts
[ec2-user@ip-172-31-11-92 scripts]$ 
```

i-026b8167f7b1b4335 (developer)
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92

2) The next one uploads the code in vi appspec.yaml!

```
version: 0.0
os: linux
files:
  - source: /index.html
    destination: /var/www/html/
hooks:
  BeforeInstall:
  - location: scripts/httpd_install.sh
    timeout: 300
    runs: root
  - location: scripts/httpd_start.sh
    timeout: 300
    runs: root
ApplicationStop:
  - location: scripts/httpd_stop.sh
    timeout: 300
    runs: root
```

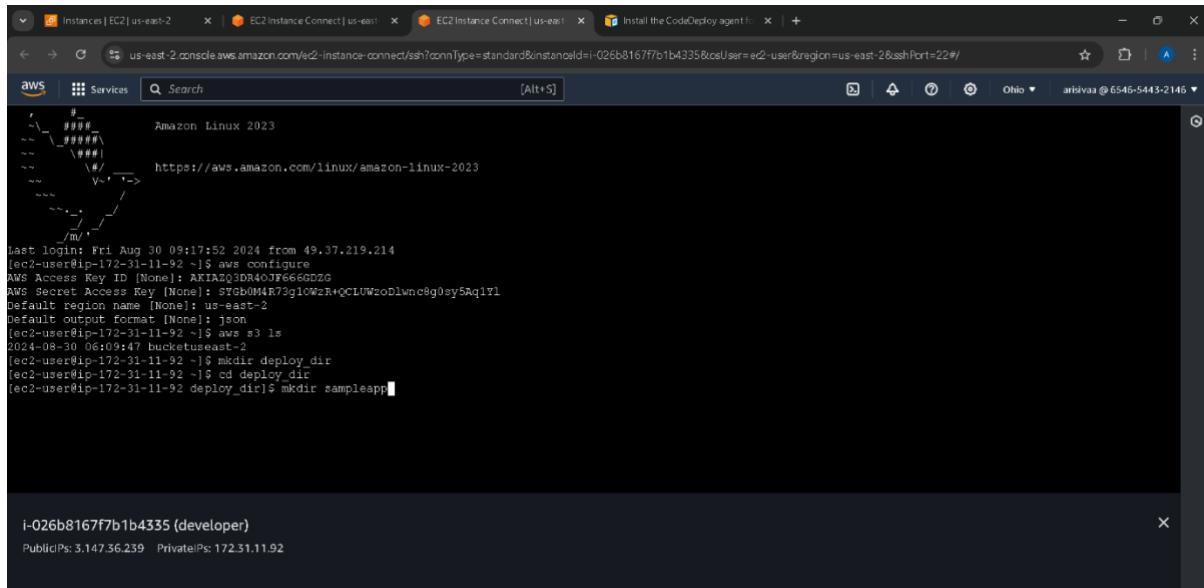
i-026b8167f7b1b4335 (developer)
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92

:wq!

Step 7: Create a scripts directory

```
[ec2-user@ip-172-31-11-92 sampleapp]$ mkdir scripts
[ec2-user@ip-172-31-11-92 sampleapp]$ cd scripts
[ec2-user@ip-172-31-11-92 scripts]$ █
```

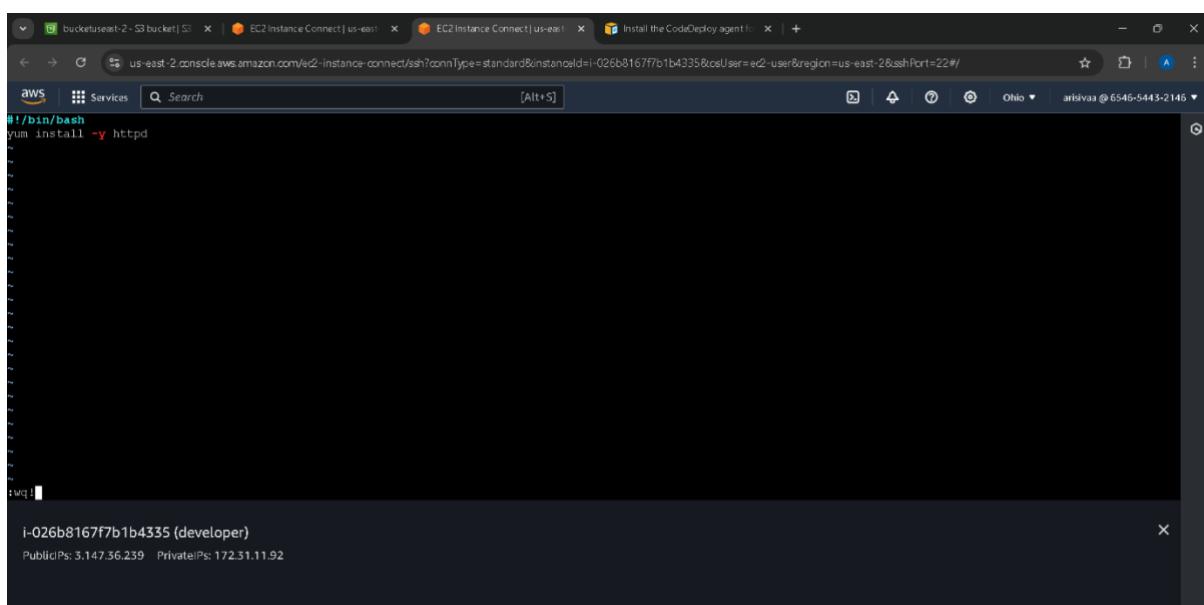
The first one is `httpd_install.sh`



```
aws Services Search [Alt+S]
Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

Last login: Fri Aug 30 09:17:52 2024 from 49.37.219.214
[ec2-user@ip-172-31-11-92 ~]$ aws configure
AWS Access Key ID [None]: AXIAQ3DR40JF666GDZG
AWS Secret Access Key [None]: STGb0M4RA73g10WzR+QCIUWzoDlwmc0g0sy5Aq1Vl
Default Region Name [None]: us-east-2
Default Output Format [None]: json
[ec2-user@ip-172-31-11-92 ~]$ s3 ls
2024-08-30 06:09:47 bucketuseast-2
[ec2-user@ip-172-31-11-92 ~]$ mkdir deploy_dir
[ec2-user@ip-172-31-11-92 ~]$ cd deploy_dir
[ec2-user@ip-172-31-11-92 deploy_dir]$ mkdir sampleapp█

I-026b8167f7b1b4335 (developer)
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92
```

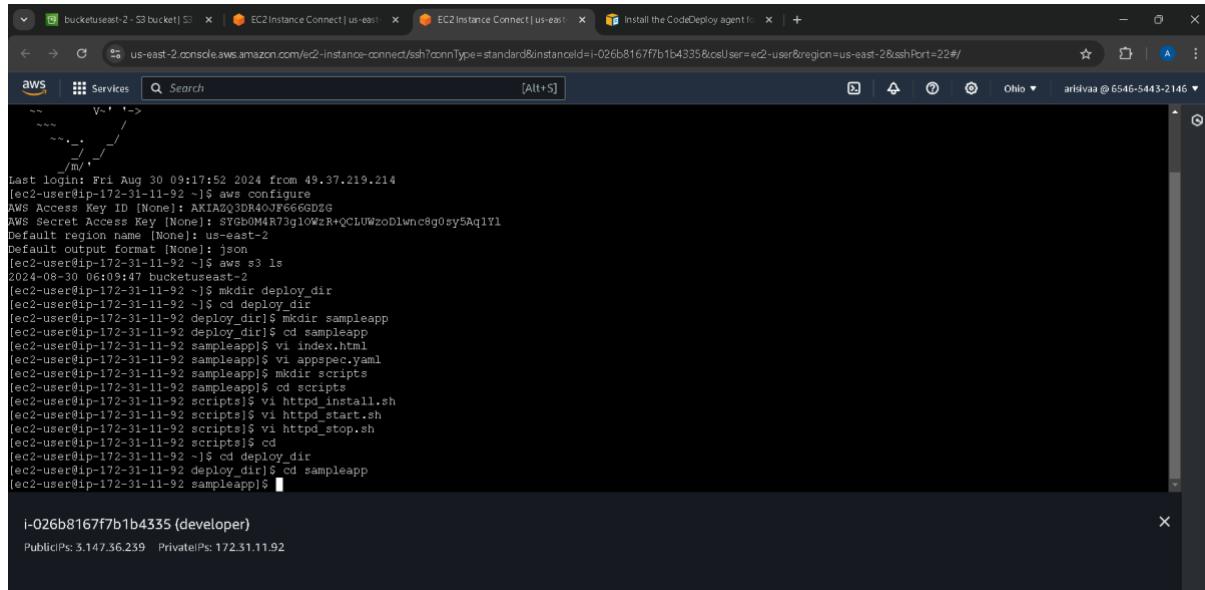


```
#!/bin/bash
yum install -y httpd
[ec2-user@ip-172-31-11-92 ~]$ █

I-026b8167f7b1b4335 (developer)
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92
```

:wq!

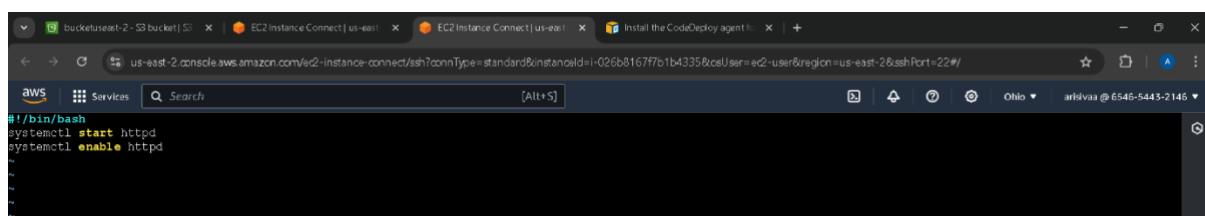
The second one is **httpd_start.sh**



```
V~` ->
`````/
````_`/
```/`m/
Last login: Fri Aug 30 09:17:52 2024 from 49.37.219.214
[ec2-user@ip-172-31-11-92 ~]$ aws configure
AWS Access Key ID [None]: AKIAZQ3DR4OJF66GDZG
AWS Secret Access Key [None]: STGb0M4R73g10WzR+qCLUWzoD1wnc8g0sy5Aq1Yl
Default region name [None]: us-east-2
Default output format [None]: json
[ec2-user@ip-172-31-11-92 ~]$ aws s3 ls
2024-08-30 06:09:47 bucketuseast-2
[ec2-user@ip-172-31-11-92 ~]$ mkdir deploy_dir
[ec2-user@ip-172-31-11-92 ~]$ cd deploy_dir
[ec2-user@ip-172-31-11-92 deploy_dir]$ mkdir sampleapp
[ec2-user@ip-172-31-11-92 deploy_dir]$ cd sampleapp
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
[ec2-user@ip-172-31-11-92 sampleapp]$ vi appspec.yaml
[ec2-user@ip-172-31-11-92 sampleapp]$ chmod 755 scripts
[ec2-user@ip-172-31-11-92 sampleapp]$ cd scripts
[ec2-user@ip-172-31-11-92 scripts]$ vi httpd_install.sh
[ec2-user@ip-172-31-11-92 scripts]$ vi httpd_start.sh
[ec2-user@ip-172-31-11-92 scripts]$ vi httpd_stop.sh
[ec2-user@ip-172-31-11-92 scripts]$ cd ..
[ec2-user@ip-172-31-11-92 deploy_dir]$ cd ..
[ec2-user@ip-172-31-11-92 sampleapp]$
```

i-026b8167f7b1b4335 (developer)

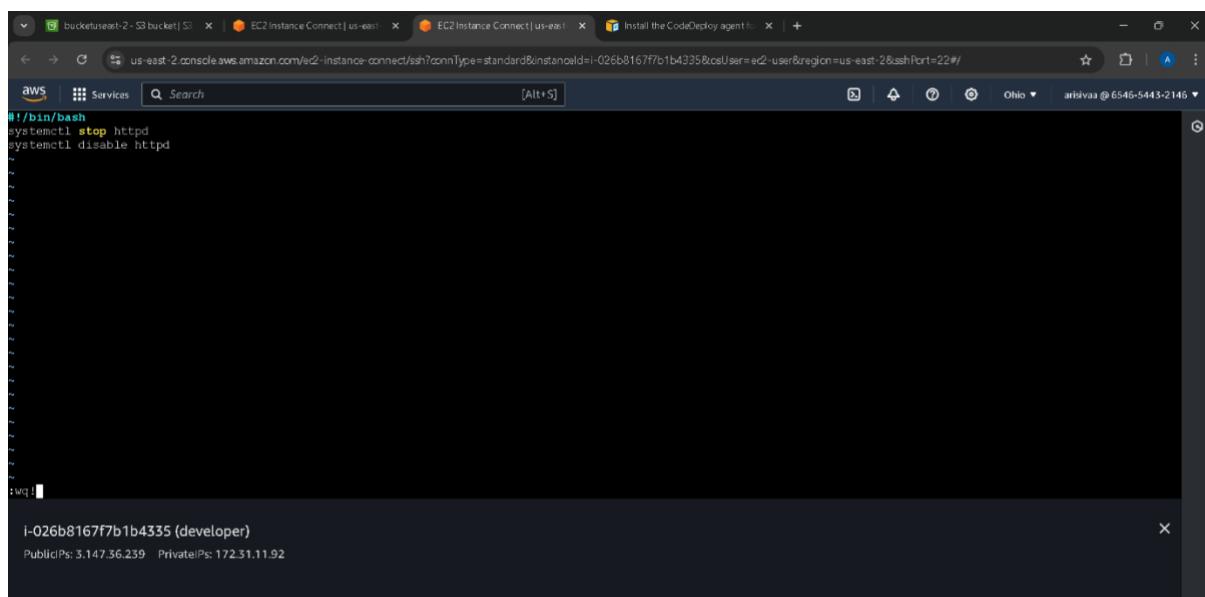
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92



```
#!/bin/bash
systemctl start httpd
systemctl enable httpd
```

:wq!

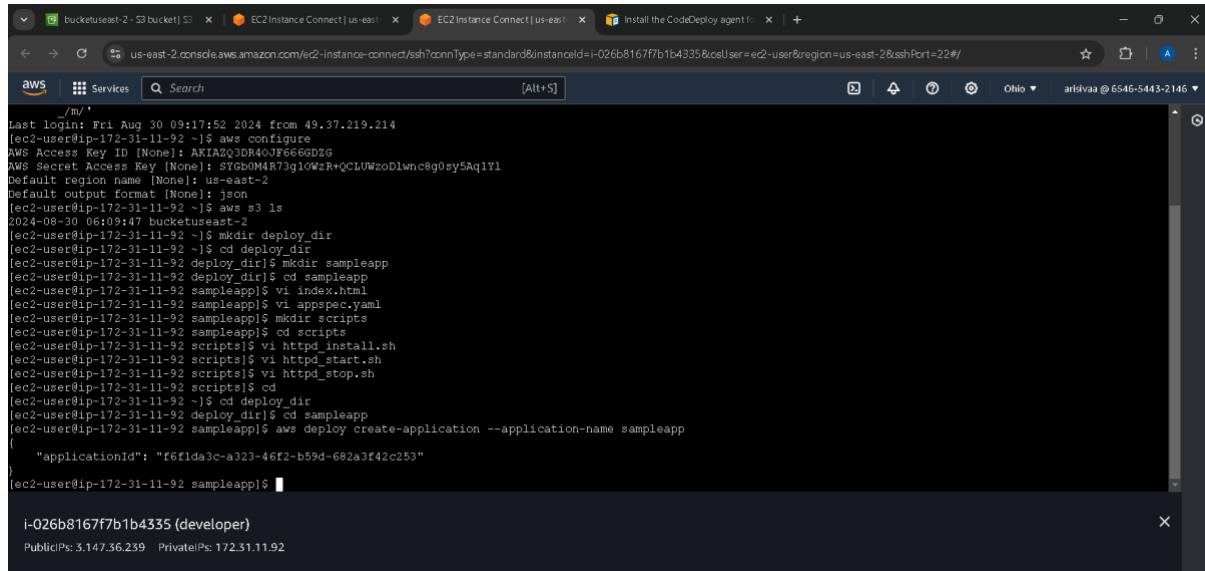
The third one is **httpd\_stop.sh**



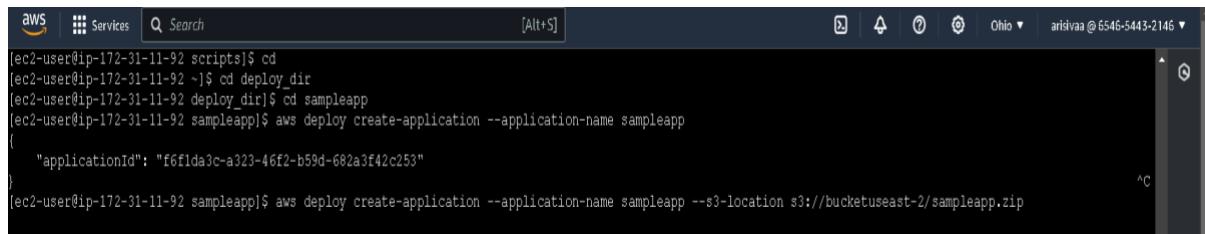
```
#!/bin/bash
systemctl stop httpd
systemctl disable httpd
```

:wq!

## Step 7: Create the code-deploy application and push the code to the s3 bucket developer machine

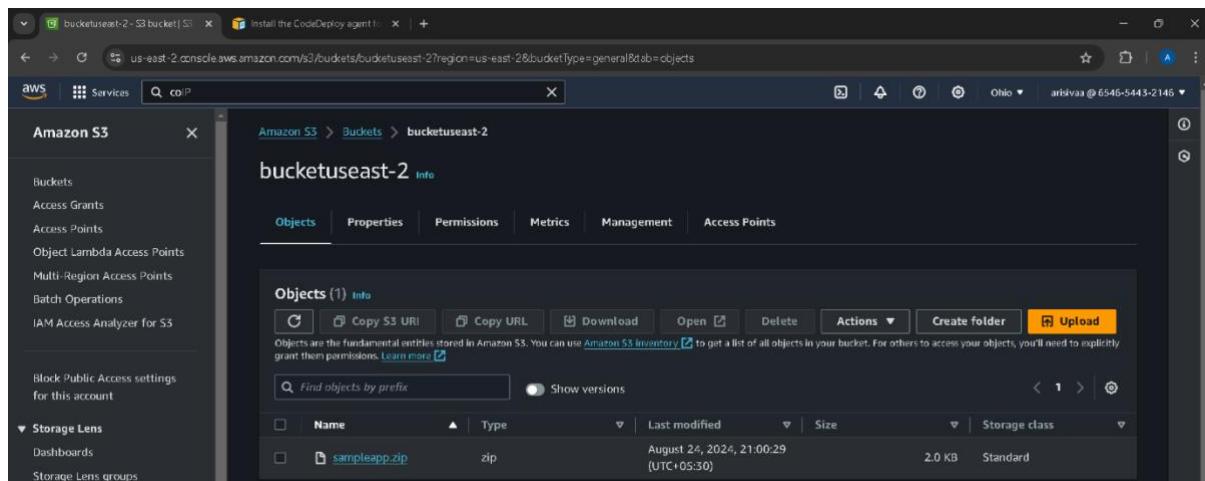


```
aws services search [Alt+S]
>Last login: Fri Aug 30 09:17:52 2024 from 49.37.219.214
[ec2-user@ip-172-31-11-92 ~]$ aws configure
AWS Access Key ID [None]: AKIAQ23DR40JF666ZG
AWS Secret Access Key [None]: stGB0M4R73gIOWzK+qCLUWzcDlwnc8gOsy5Aq1Y1
Default region name [None]: us-east-2
Default output format [None]: json
[ec2-user@ip-172-31-11-92 ~]$ aws s3 ls
2024-08-30 06:09:47 0 bucketuseast-2
[ec2-user@ip-172-31-11-92 ~]$ cd deploy_dir
[ec2-user@ip-172-31-11-92 ~]$ cd deploy_dir
[ec2-user@ip-172-31-11-92 deploy_dir]$ mkdir sampleapp
[ec2-user@ip-172-31-11-92 deploy_dir]$ cd sampleapp
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
[ec2-user@ip-172-31-11-92 sampleapp]$ vi appspec.yaml
[ec2-user@ip-172-31-11-92 sampleapp]$ mkdir scripts
[ec2-user@ip-172-31-11-92 scripts]$ cd scripts
[ec2-user@ip-172-31-11-92 scripts]$ vi httpd.install.sh
[ec2-user@ip-172-31-11-92 scripts]$ vi httpd_start.sh
[ec2-user@ip-172-31-11-92 scripts]$ vi httpd_stop.sh
[ec2-user@ip-172-31-11-92 scripts]$ cd ..
[ec2-user@ip-172-31-11-92 ~]$ cd deploy_dir
[ec2-user@ip-172-31-11-92 deploy_dir]$ cd sampleapp
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy create-application --application-name sampleapp
{
 "applicationId": "f6ff1da3c-a323-46f2-b59d-682a3f42c253"
}
[ec2-user@ip-172-31-11-92 sampleapp]$ i-026b8167f7b1b4335 (developer)
PublicIPs: 5.147.36.239 PrivateIPs: 172.31.11.92
```



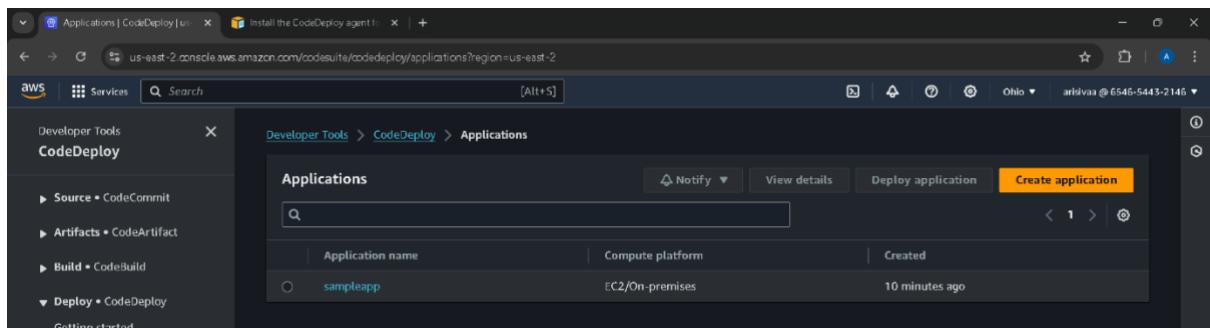
```
[ec2-user@ip-172-31-11-92 scripts]$ cd
[ec2-user@ip-172-31-11-92 ~]$ cd deploy_dir
[ec2-user@ip-172-31-11-92 deploy_dir]$ cd sampleapp
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy create-application --application-name sampleapp
{
 "applicationId": "f6ff1da3c-a323-46f2-b59d-682a3f42c253"
}
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy create-application --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip
```

### a) create a bucket name called bucket bucketuseast-2 – make it public policy



The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with options like Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, and Block Public Access settings. The main area shows the 'bucketuseast-2' bucket details. Under the 'Objects' tab, there is one object listed: 'sampleapp.zip' (Type: zip, Last modified: August 24, 2024, 21:00:29 (UTC+05:30), Size: 2.0 KB, Storage class: Standard). There are also tabs for Properties, Permissions, Metrics, Management, and Access Points.

Check the Codedeploy in the Console Level in the GUI



Next, create a s3 command

```
aws deploy push --application-name sampleapp --s3-location
s3://bucketuseast-2/sampleapp.zip
```

```
Unknown options: create-application
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy push --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip

/home/ec2-user/deploy_dir/sampleapp/appspec.yml was not found
[ec2-user@ip-172-31-11-92 sampleapp]$ mv appspec.yaml appspec.yml
[ec2-user@ip-172-31-11-92 sampleapp]$
```

i-026b8167f7b1b4335 (developer)  
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92

Successfully create a s3 command.

```
Unknown options: --s3-location, s3://bucketuseast-2/sampleapp.zip
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy push create-application --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip

Unknown options: create-application
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy push create-application --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip

Unknown options: create-application
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy push --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip

/home/ec2-user/deploy_dir/appspec.yml was not found
[ec2-user@ip-172-31-11-92 sampleapp]$ mv appspec.yaml appspec.yml
[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy push --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:
 aws help
 aws <command> help
 aws <command> <subcommand> help

aws: error: the following arguments are required: --application-name

[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy push --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip
To deploy with this revision, run:
aws deploy create-deployment --application-name sampleapp --s3-location bucket=bucketuseast-2,key=sampleapp.zip,bundleType=zip,eTag=6551deeed036d617f4c17a6968a105
--deployment-group-name <deployment-group-name> --deployment-config-name <deployment-config-name> --description <description>
[ec2-user@ip-172-31-11-92 sampleapp]$
```

i-026b8167f7b1b4335 (developer)  
PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92

## **Step 8: Create a deployment group.**

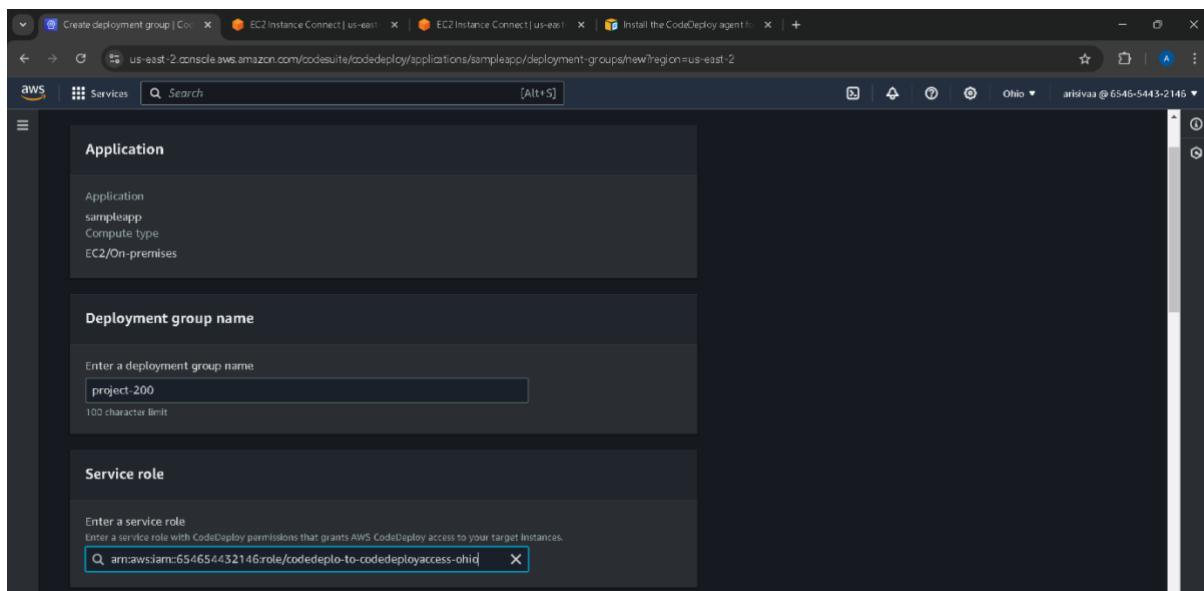
**Application:**

**Sampleapp**

**Compute type:**

**EC2/on premises**

### **Create a deployment group name and service role**



**The next one choose how to deploy your application ( in place or blue/green )**

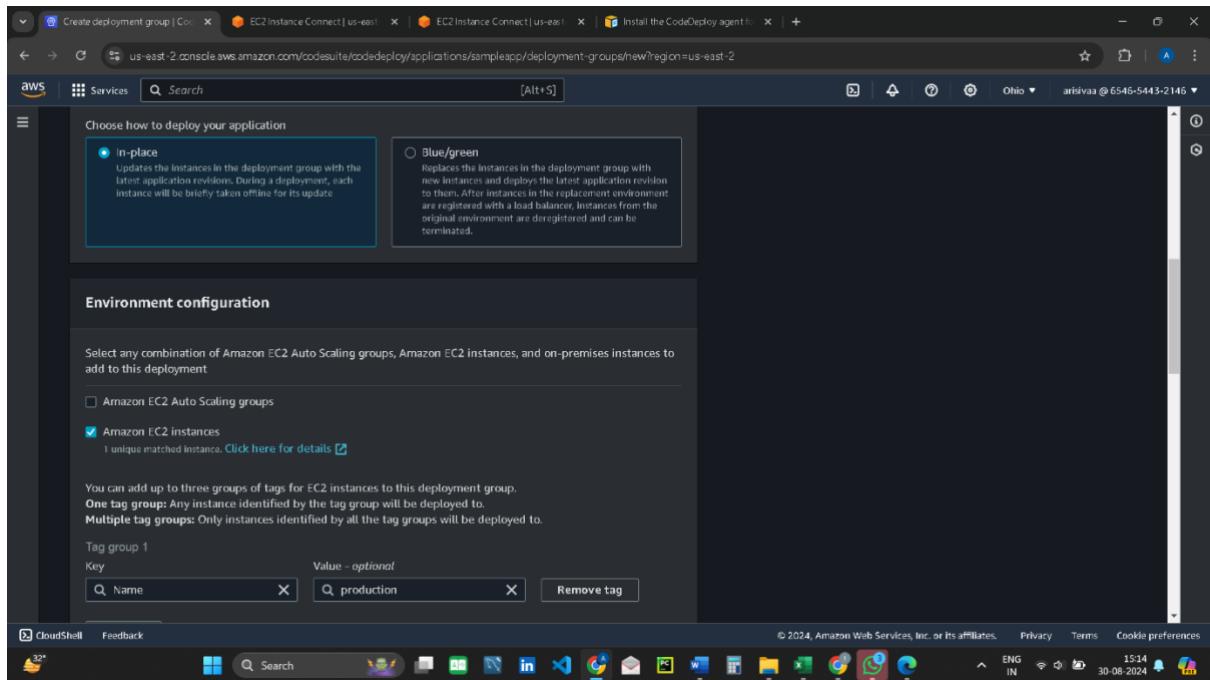
Choose the environment configuration

**Amazon EC2 instances**

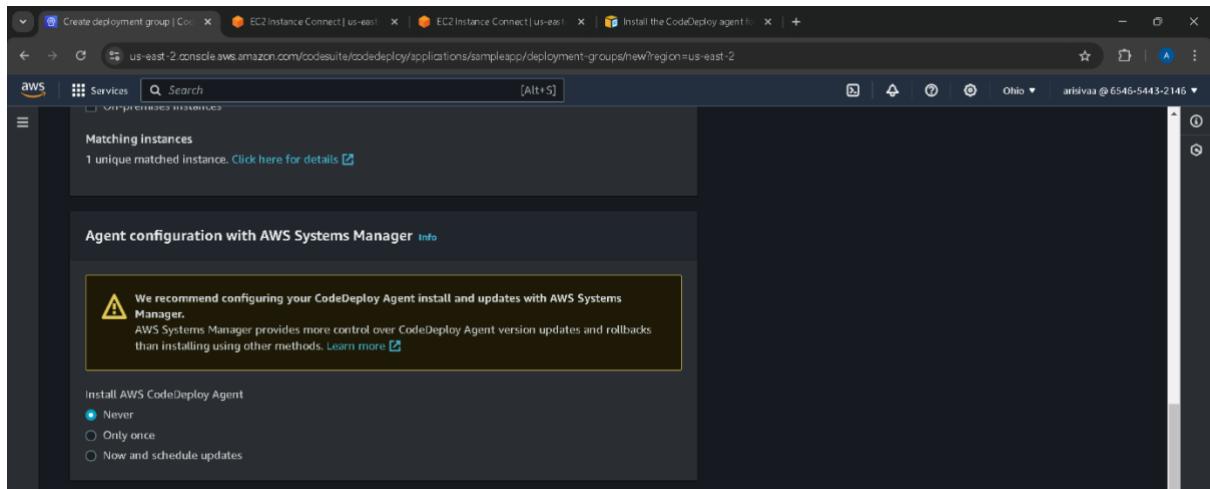
**Tag group:**

**Key: name**

**Value: production**



## Install AWS code-deploy agent **Never**



**Successfully create a deployment group**

The screenshot shows the AWS CodeDeploy console. On the left, a sidebar menu under 'Developer Tools' has 'CodeDeploy' selected. The main area displays a green success message: 'Success Deployment group created'. Below this, the 'Deployment group details' section shows the following information:

Deployment group name	Application name	Compute platform
project-200	sampleapp	EC2/On-premises
Deployment type	Service role ARN	Deployment configuration
In-place	arn:aws:iam::654654432146:role/codedeploy-to-codedeploy-access-ohio	CodeDeployDefault.AllAtOnce
Rollback enabled	Agent update scheduler	
False	Learn to schedule update in AWS Systems Manager	

I have created a code deployment.

The screenshot shows the 'Create deployment' wizard. The first step, 'Deployment settings', is displayed. The application is set to 'sampleapp' and the deployment group is set to 'project-200'. Other settings include:

- Compute platform: EC2/On-premises
- Deployment type: In-place
- Managed hook execution role: Not explicitly shown, but a note indicates it's used for actions.
- Revision type: My application is stored in Amazon S3 (selected)
- Revision location: s3://bucketuseast-2/sampleapp.zip?eTag=6551deeed036d617f4c17a6968; X

Successfully create code deployment.

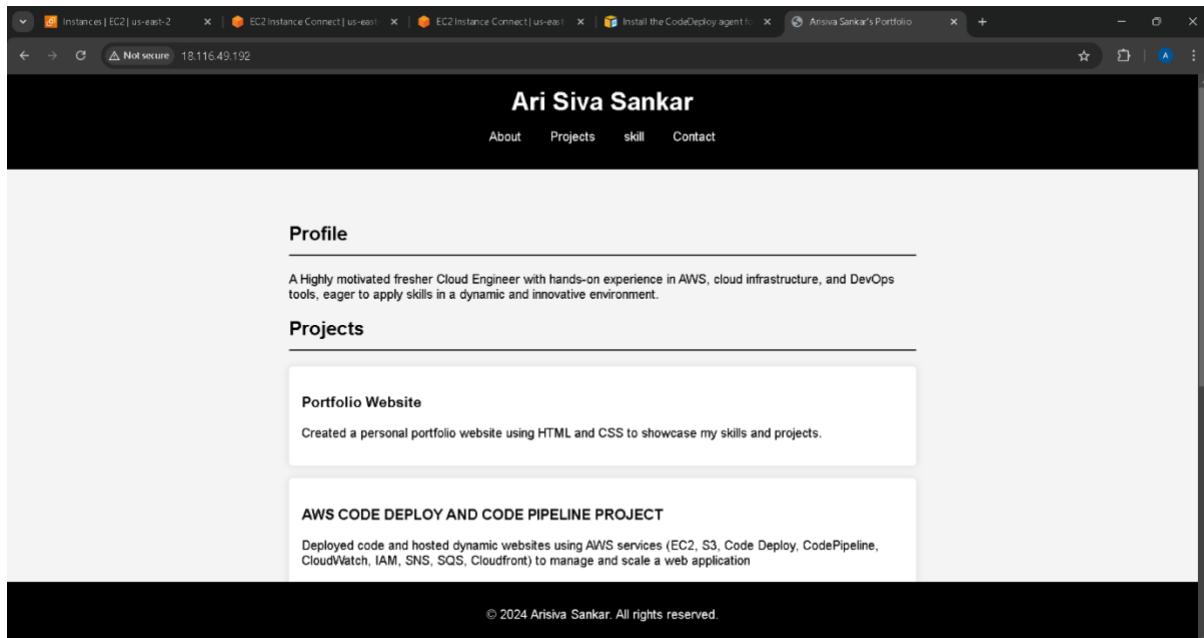
The screenshot shows the deployment status for deployment 'd-ZK4MAX4T6'. The status bar at the top says 'Success Deployment created'. The main area shows the deployment status:

Deployment status: Installing application on your instances (100%)

1 of 1 Instances updated: Succeeded

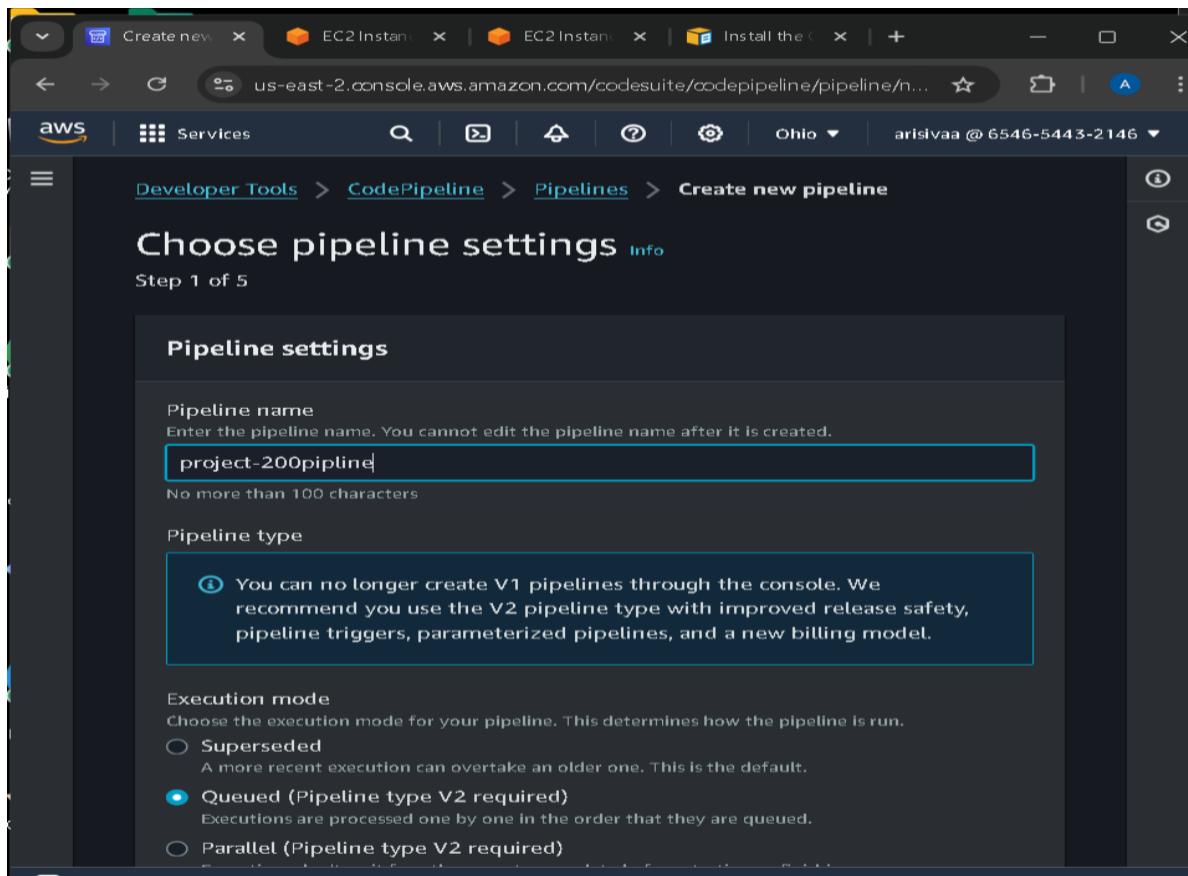
Buttons: Copy deployment, Retry deployment

The next step copy the production machine's IP address and paste it Chrome browser next run the browser to check the output.



## Step 8: Create a code pipeline

- 1) create a code pipeline name and Choose the pipeline type.



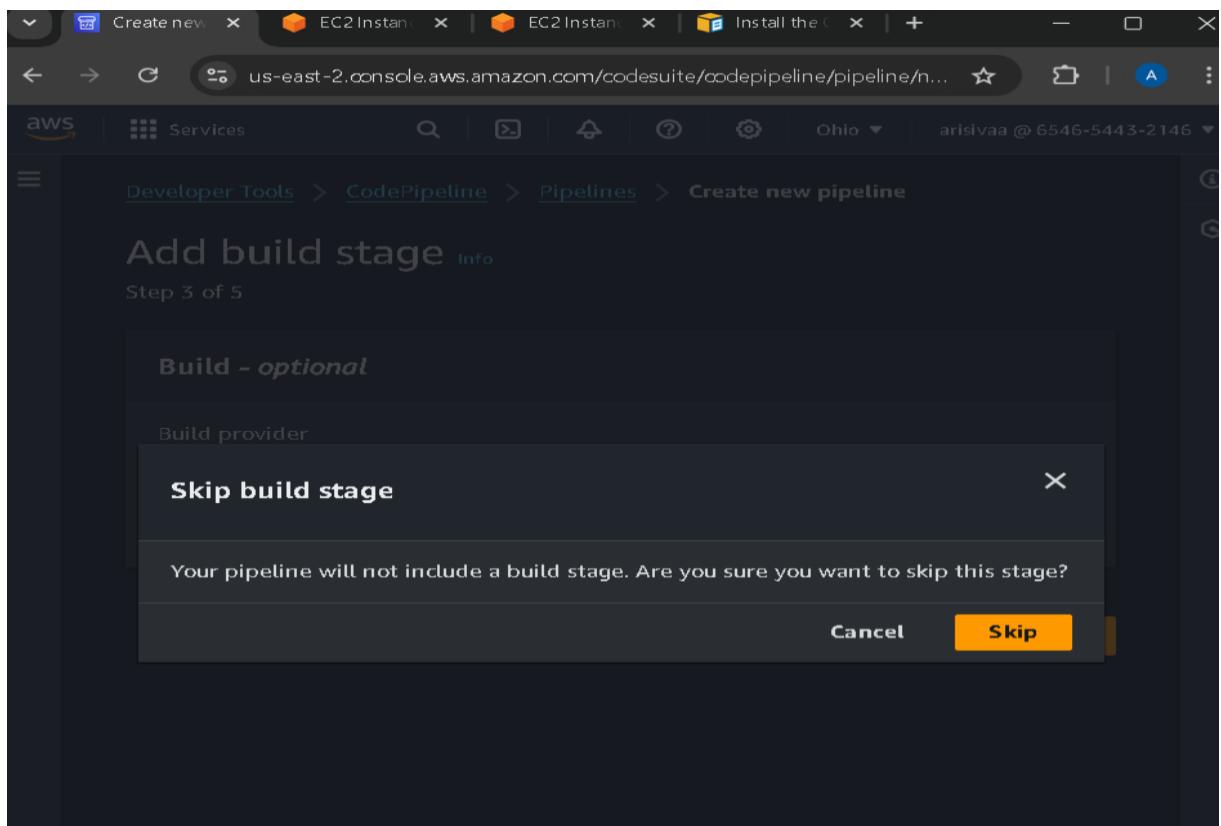
- 2) choose the excited service role and role arn.

The screenshot shows the AWS CodePipeline setup interface. Under 'Execution mode', 'Queued (Pipeline type V2 required)' is selected. Under 'Service role', 'New service role' is selected, with the role name 'AWSCodePipelineServiceRole-us-east-2-project-200pipeline' entered. A checked checkbox allows AWS CodePipeline to create a service role. The interface includes tabs for 'Create new', 'EC2 Instance', 'EC2 Instance', and 'Install the'.

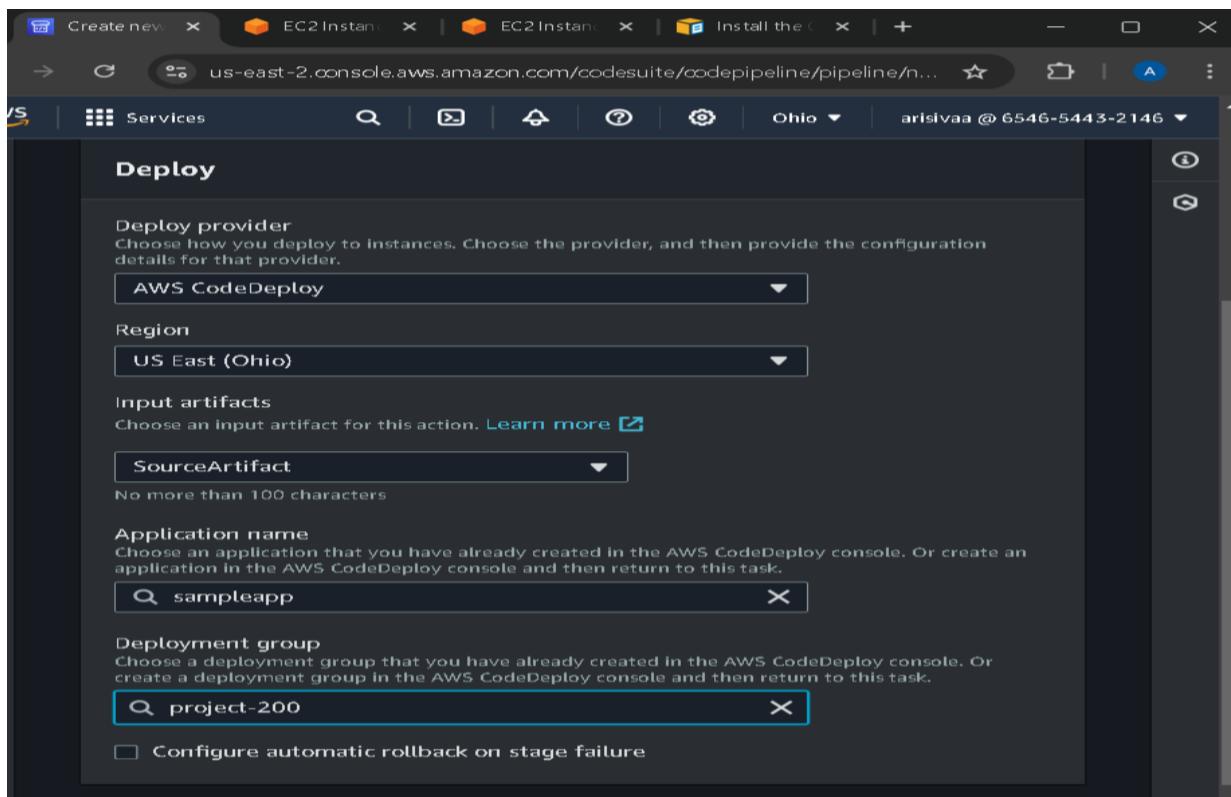
- 3) add source stage choose the bucket and select the object next one create a key value and choose detection option.

The screenshot shows the 'Source provider' configuration screen. 'Amazon S3' is selected as the provider. The 'Bucket' field contains 'bucketuseast-2'. The 'S3 object key' field contains 'sampleapp'. In the 'Change detection options' section, 'AWS CodePipeline' is selected. The interface includes tabs for 'Create new', 'EC2 Instance', 'EC2 Instance', and 'Install the'.

4) skip build stage.



5) choose the deploy porivder and region. the second one select the input artifacts and then select the applicationnam and deployment group



## 6) Successfully create a codepipeline.

The last one enables the s3 versioning and then checks the code pipeline.

A screenshot of the AWS CodePipeline console. At the top, a green success message box says "Congratulations! The pipeline project-200pipline has been created." Below it, a button says "Create a notification rule for this pipeline". The navigation path is "Developer Tools > CodePipeline > Pipelines > project-200pipline". The pipeline name is "project-200pipline". It shows a "Release change" button, "Pipeline type: V2", and "Execution mode: QUEUED". A tooltip message states: "This pipeline has a source (Source) that is configured for polling. Migrate (update) your pipeline to the recommended event-based mechanism for change detection. For details, see the migration guide." Below this, there's a "Source" step with "Source" and "Amazon S3" options. On the right, there are two small circular icons.

A screenshot of a web browser displaying a personal portfolio website. The page content includes:

```
<p>Created a personal portfolio website using HTML and CSS to showcase my skills and projects.</p>
<h3>AWS CODE DEPLOY AND CODE PIPELINE PROJECT</h3>
<h3>Skills</h3>
<h3>AWS (amazon web services)</h3>
<h3>Skills</h3>
<h3>Skills</h3>
<h3>Skills</h3>
<h3>python</h3>
<h3>Contact</h3>

<p>If you'd like to get in touch, please email me at arisiva061@gmail.com.</p>
<footer>
 <p>© 2024 Arisiva Sankar. All rights reserved.</p>
</footer>
```

At the bottom, it shows the developer information: "I-026b8167f7b1b4335 (developer)" and "Public IPs: 3.147.56.239 Private IPs: 172.31.11.92".

The next step edit the vi file **vi index.html**

**Zip -r ..sampleapp.zip**

**Ls**

**Cd ..**

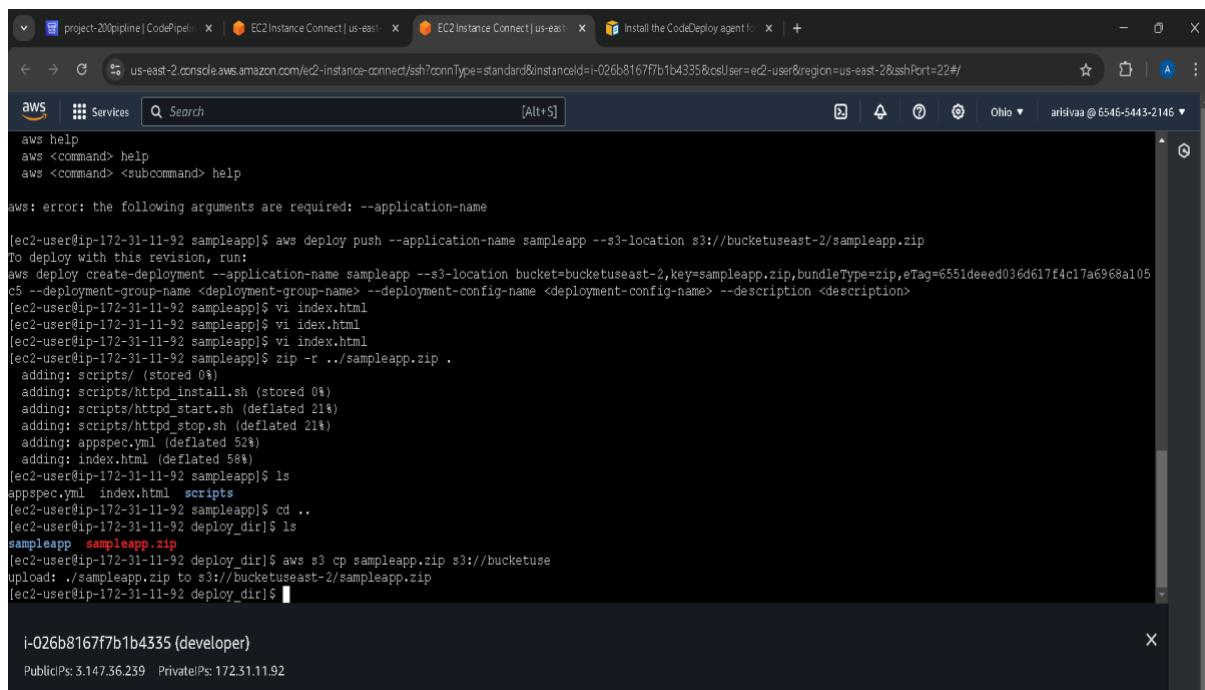
**Ls**

**Cd ..**

**Ls**

**You see the sampleapp.zip**

**The next time, type this command. aws s3 cp sampleapp.zip s3://bucketuseast-2**



```
aws help
aws <command> help
aws <command> <subcommand> help

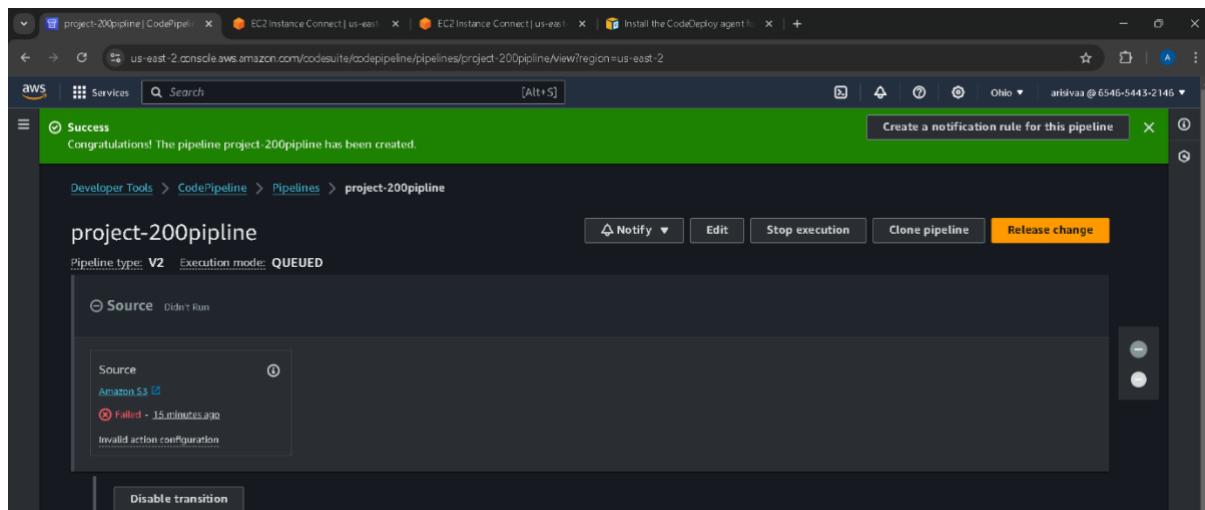
aws: error: the following arguments are required: --application-name

[ec2-user@ip-172-31-11-92 sampleapp]$ aws deploy push --application-name sampleapp --s3-location s3://bucketuseast-2/sampleapp.zip
To deploy with this revision, run:
aws deploy create-deployment --application-name sampleapp --s3-location bucket=bucketuseast-2,key=sampleapp.zip,bundleType=zip,eTag=6551deeed036d617f4c17a6968a105
--deployment-group-name <deployment-group-name> --deployment-config-name <deployment-config-name> --description <description>
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
[ec2-user@ip-172-31-11-92 sampleapp]$ vi index.html
[ec2-user@ip-172-31-11-92 sampleapp]$ zip -r ..sampleapp.zip .
adding: scripts/ (stored 0%)
adding: scripts/httpd_install.sh (stored 0%)
adding: scripts/httpd_start.sh (deflated 21%)
adding: scripts/httpd_stop.sh (deflated 21%)
adding: appspec.yml (deflated 52%)
adding: index.html (deflated 58%)
[ec2-user@ip-172-31-11-92 sampleapp]$ ls
appspec.yml index.html scripts
[ec2-user@ip-172-31-11-92 sampleapp]$ cd ..
[ec2-user@ip-172-31-11-92 deploy_dir]$ ls
sampleapp sampleapp.zip
[ec2-user@ip-172-31-11-92 deploy_dir]$ aws s3 cp sampleapp.zip s3://bucketuseast-2
upload: ./sampleapp.zip to s3://bucketuseast-2/sampleapp.zip
[ec2-user@ip-172-31-11-92 deploy_dir]$
```

i-026b8167f7b1b4335 (developer)

PublicIPs: 3.147.36.239 PrivateIPs: 172.31.11.92

**S3 invalid action configuration.**



## Edit bucket versioning. enable

The screenshot shows the 'Edit Bucket Versioning' page in the AWS S3 console. The 'Bucket Versioning' section is displayed, with the 'Enable' radio button selected. A callout box highlights the note: 'After enabling Bucket Versioning, you might need to update your lifecycle rules to manage previous versions of objects.' Below this, the 'Multi-factor authentication (MFA) delete' section is visible, showing it is 'Disabled'. At the bottom right are 'Cancel' and 'Save changes' buttons.

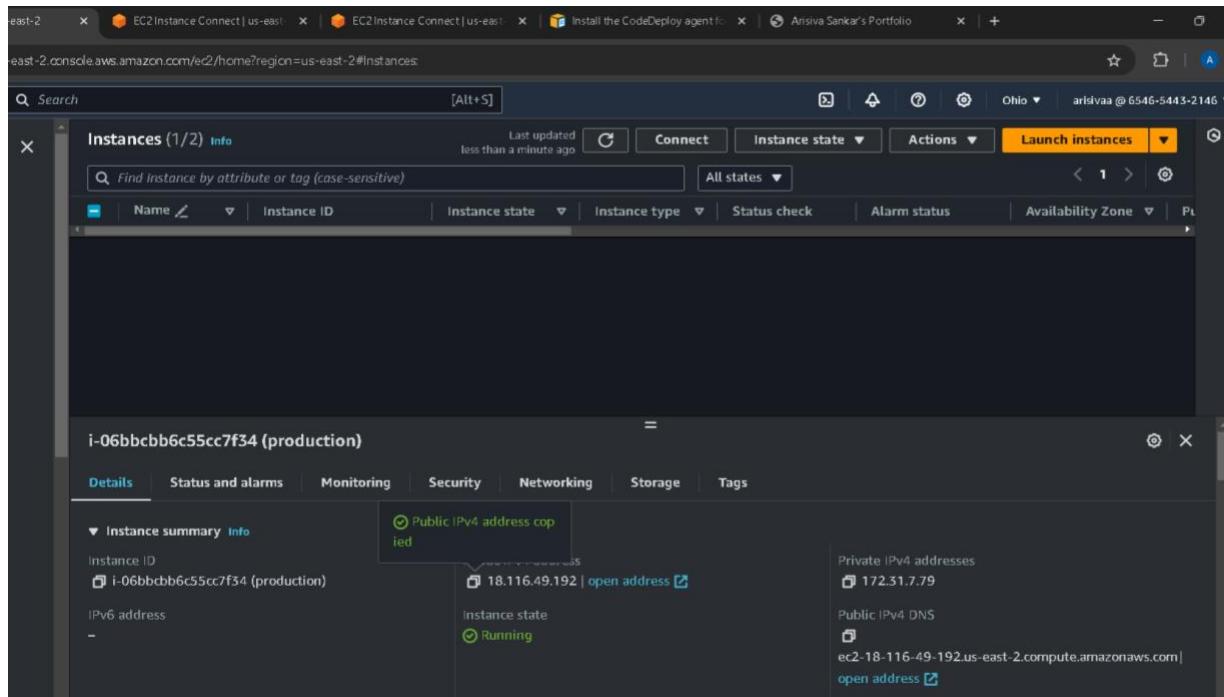
Successfully edited bucket versioning.

The screenshot shows a green success message box stating 'Successfully edited Bucket Versioning' and 'To transition, archive, or delete older object versions, configure lifecycle rules for this bucket.'

Successfully finished

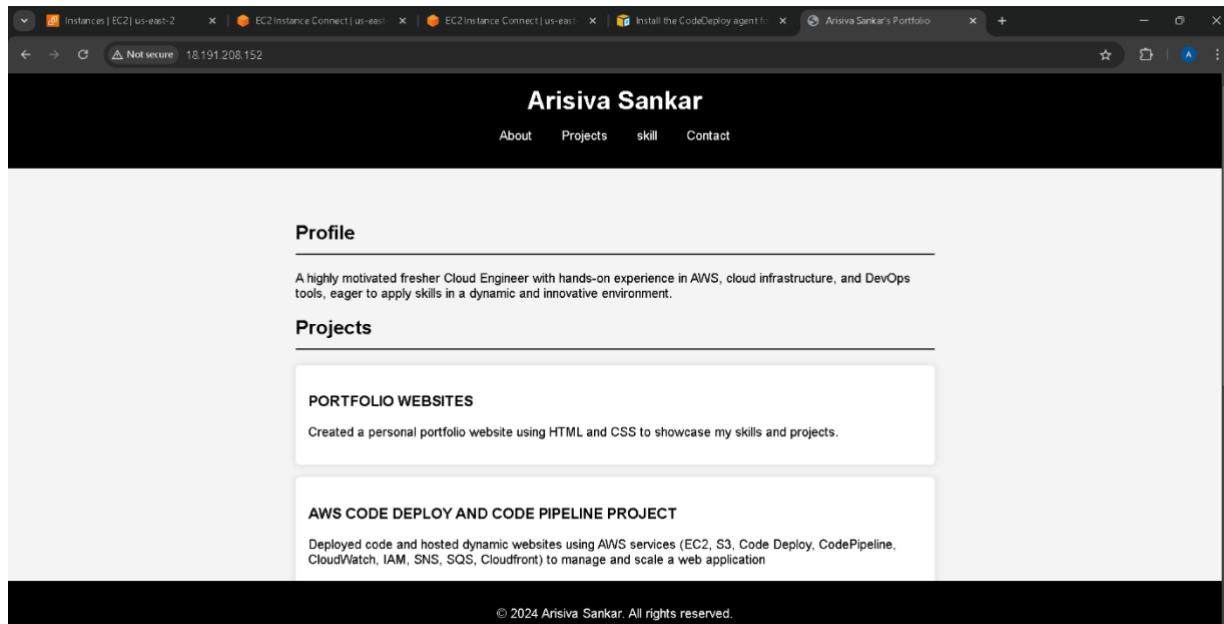
The screenshot shows the 'project-200pipeline' pipeline in the AWS CodePipeline console. The pipeline type is V2 and the execution mode is QUEUED. The pipeline consists of two stages: 'Source' and 'Deploy'. The 'Source' stage is listed as 'Succeeded' with a Pipeline execution ID of 0983a203-6650-4508-a72f-1342e79af230. The 'Deploy' stage is also listed as 'Succeeded' with a Pipeline execution ID of 0983a203-6650-4508-a72f-1342e79af230. Buttons for 'Notify', 'Edit', 'Stop execution', 'Clone pipeline', and 'Release change' are visible at the top of the pipeline view.

The next one copies the production machine IP address and pastes it into your browser.



## Output

Successfully run my portfolio website

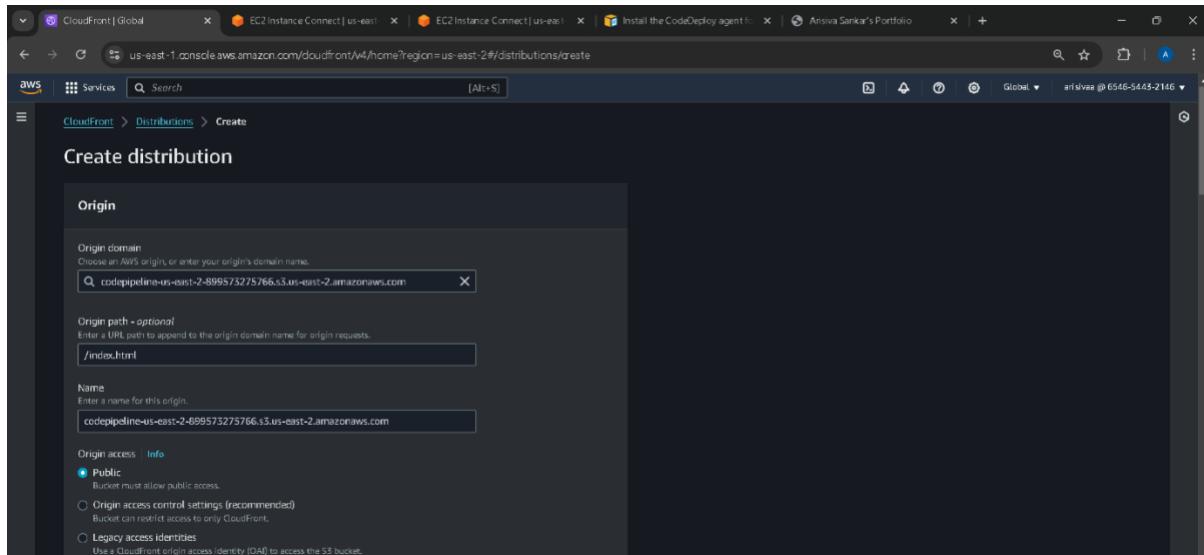


## AWS SERVICES USED:

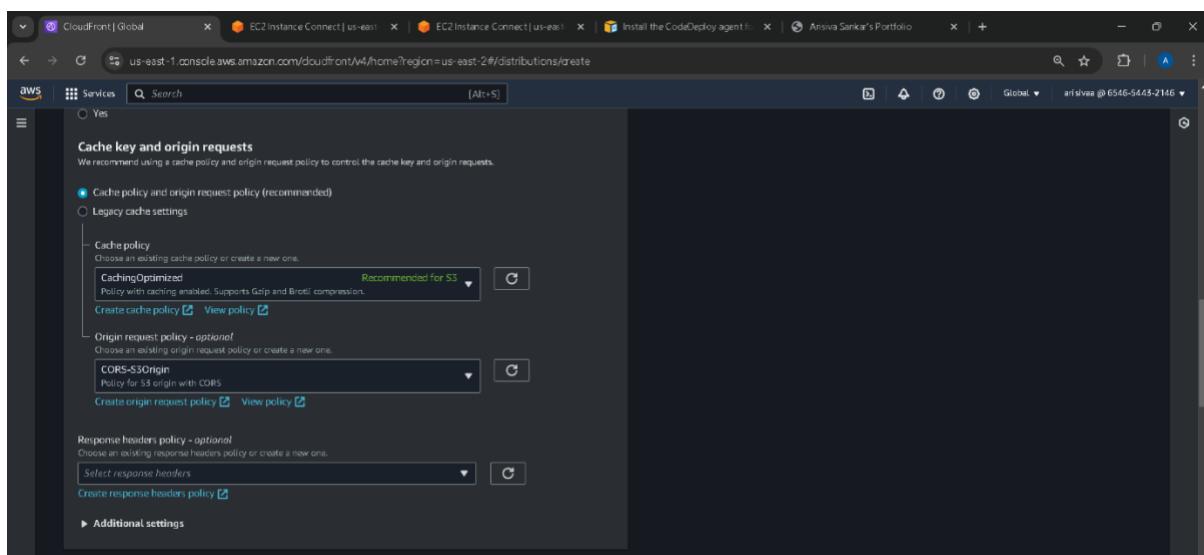
### CloudFront:

**AWS CloudFront** is a content delivery network (CDN) that speeds up the distribution of your web content by caching it at edge locations globally, reducing latency for users. It also provides DDoS protection and supports seamless integration with other AWS services.

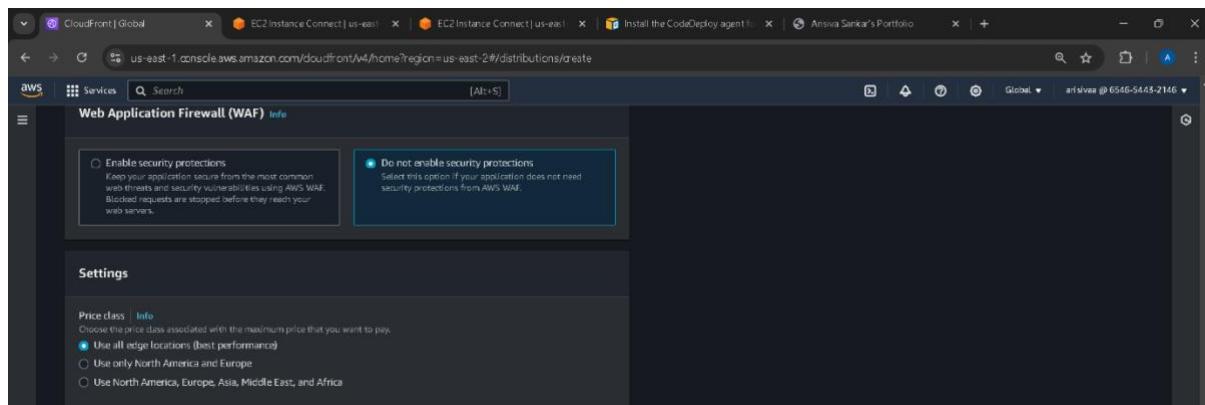
#### 1) Select the Origin and origin path then select origin access public



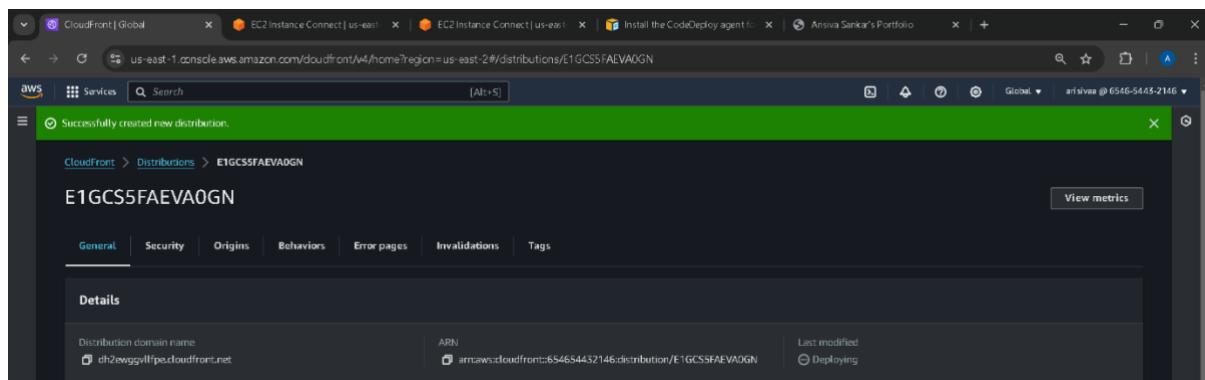
#### 2) Cache key and origin request.



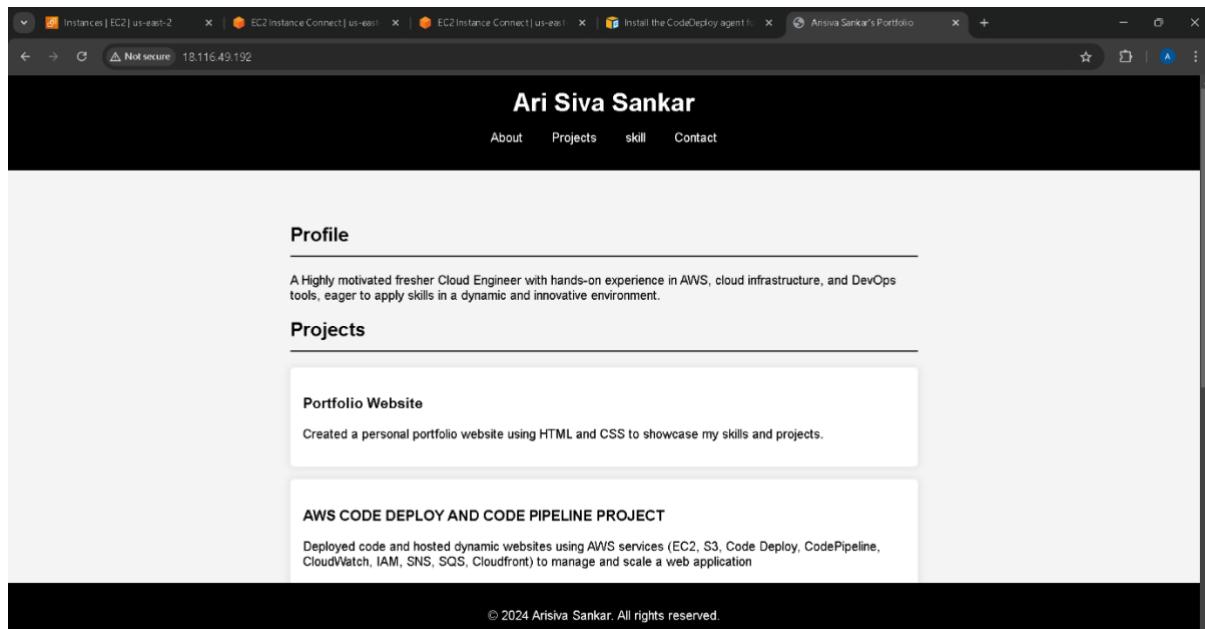
### 3) Do not enable security protections and select the use all edge location best performances



### 4) Successfully created new distribution

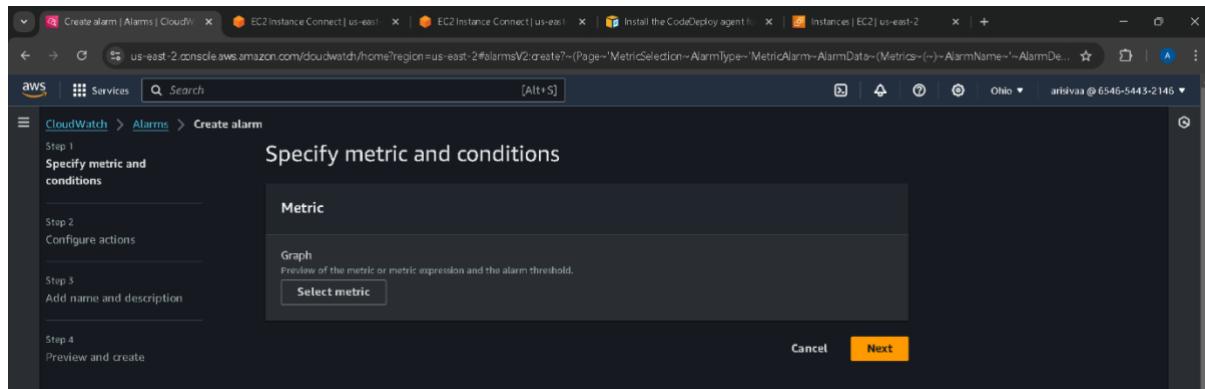


### Output:

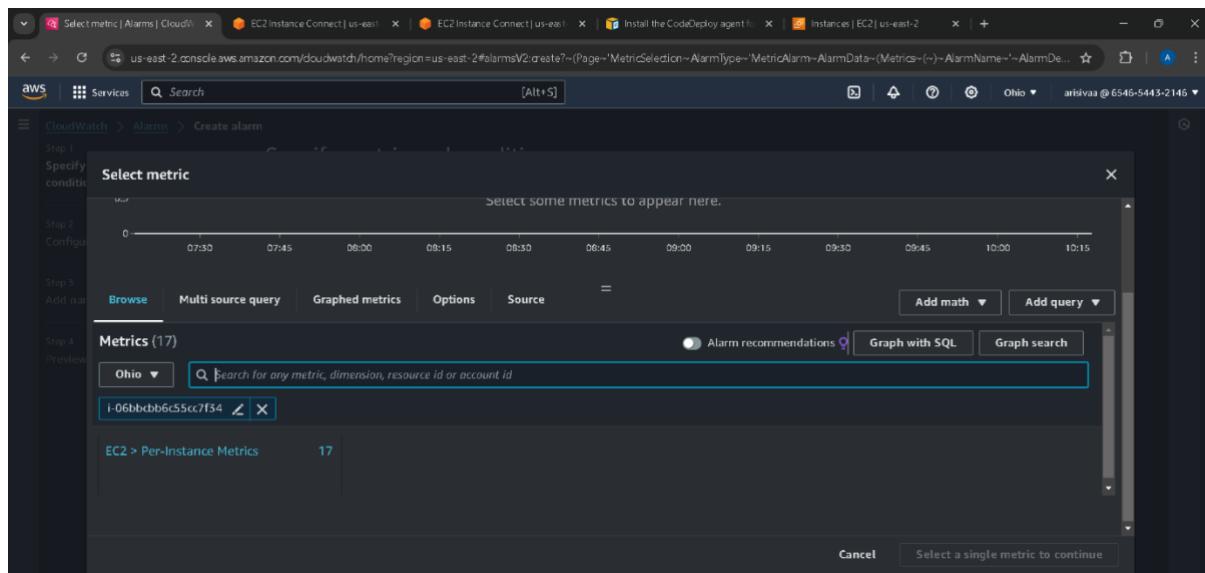


## Cloudwatch:

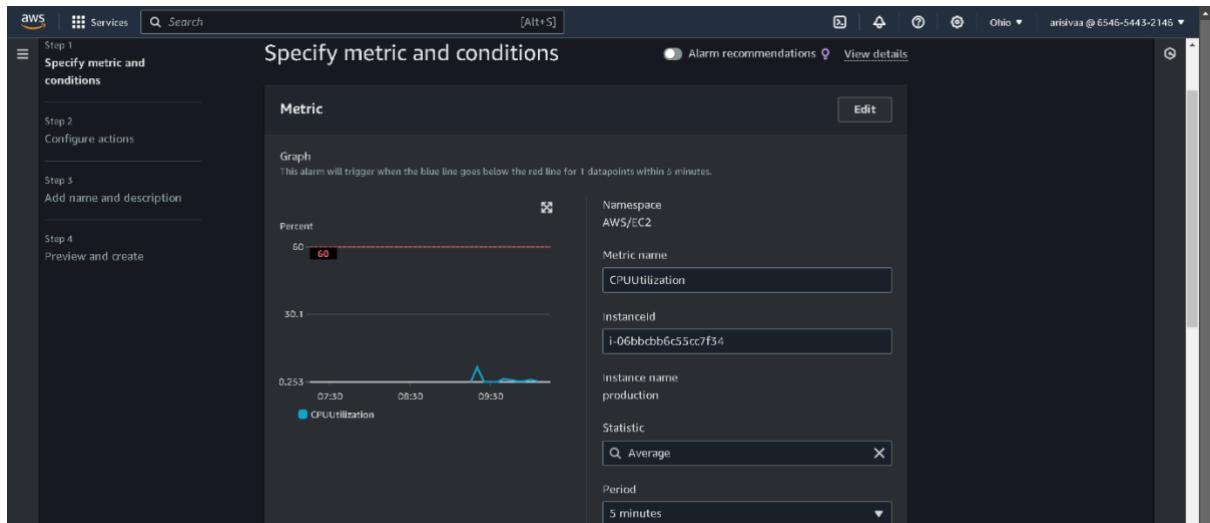
### 1) Select metric.



### 2) Pre-instance metric.



### 3) Select the metric CPU utilization.



#### 4) Select the conditions.

**Conditions**

Threshold type

Static  
Use a value as a threshold

Anomaly detection  
Use a band as a threshold

Whenever CPUUtilization is...

Define the alarm condition.

Greater  
> threshold

Greater/Equal  
>= threshold

Lower/Equal  
<= threshold

Lower  
< threshold

than...

Define the threshold value.

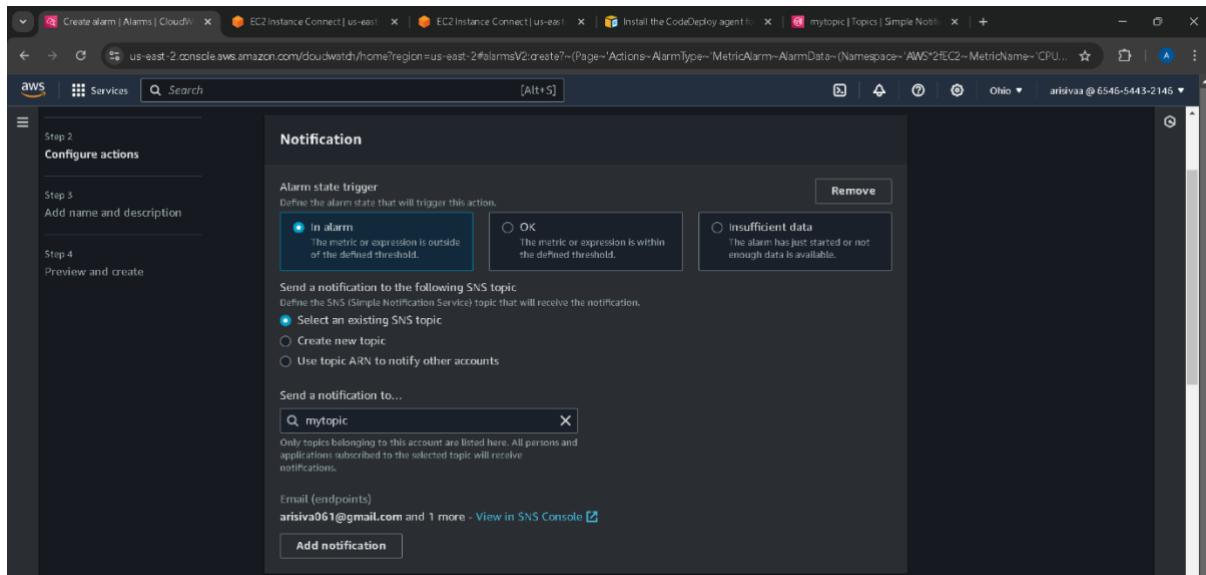
60

Must be a number

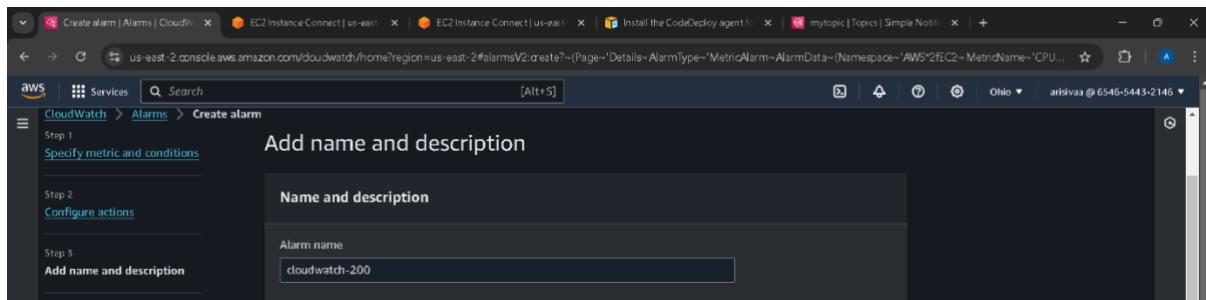
▶ Additional configuration

Cancel Next

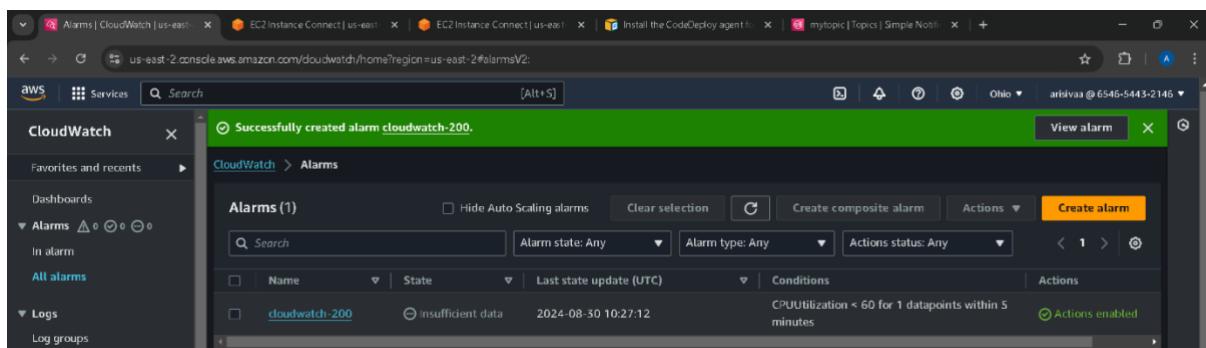
#### 5) Notification.



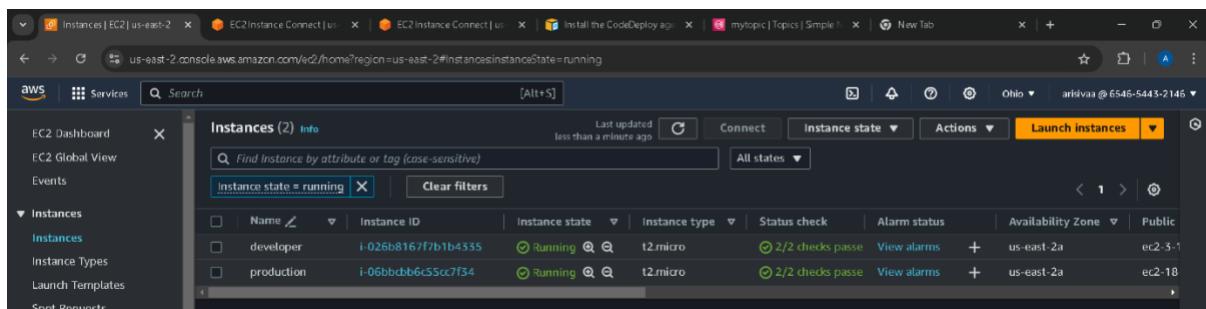
## 6) Create a alarm name.



## 7) Successfully create an alarm.



## 8) Check instances 4 or 5 times reload the page and check the cloud watch.



## 9) The check cloud watch in alarm status.

The screenshot shows the AWS CloudWatch Alarms interface. The left sidebar has 'CloudWatch' selected under 'Alarms'. The main area shows a table with one row for 'cloudwatch-200'. The 'State' column shows a red triangle icon indicating 'in alarm'. The 'Actions' column shows 'Actions enabled'.

## 10) The next check is the EC2 instances.

The screenshot shows the AWS EC2 Instances page. The left sidebar has 'Instances' selected. The main area shows a table with two rows: 'developer' and 'production'. The 'Alarm status' column for 'developer' shows '2/2 checks passed'. For 'production', it shows '1 in alarm' with a red triangle icon. The 'Actions' column for 'production' shows 'View alarms' and '+'. The 'Status check' column for 'production' shows '2/2 checks passed'.

## SNS ( SIMPLE NOTIFICATION SERVICE )

### 1) I have already created an SNS topic so I don't have any screen sorry.

The screenshot shows the AWS SNS Topics page. The left sidebar has 'Topics' selected. The main area shows a table with one row for 'mytopic'. The 'Name' column shows 'mytopic'. The 'Display name' column shows 'mytopic'. The 'ARN' column shows 'arn:aws:sns:us-east-2:654654432146:mytopic'. The 'Topic owner' column shows '654654432146'.

### 2) Next create a subscription first Email.

Select the topic ARN and protocol Email and endpoint arisiva061@gmail.com

**Create subscription**

**Details**

Topic ARN  
arn:aws:sns:us-east-2:654654432146:mytopic

Protocol  
Email

Endpoint  
An email address that can receive notifications from Amazon SNS.  
arisia061@gmail.com

ⓘ After your subscription is created, you must confirm it. [Info](#)

**Amazon SNS**

**Subscription to mytopic created successfully.**  
The ARN of the subscription is arn:aws:sns:us-east-2:654654432146:mytopic:e1ad946a-2ce4-44fd-9bb7-1041d42f1e3d.

**Subscription: e1ad946a-2ce4-44fd-9bb7-1041d42f1e3d**

**Details**

ARN arn:aws:sns:us-east-2:654654432146:mytopic:e1ad946a-2ce4-44fd-9bb7-1041d42f1e3d	Status Pending confirmation
Endpoint arisia061@gmail.com	Protocol EMAIL
Topic mytopic	Subscription Principal arn:aws:iam::654654432146:user/arisiaaa

### 3) Pending confirmation.

**Amazon SNS**

**Topics**

**mytopic**

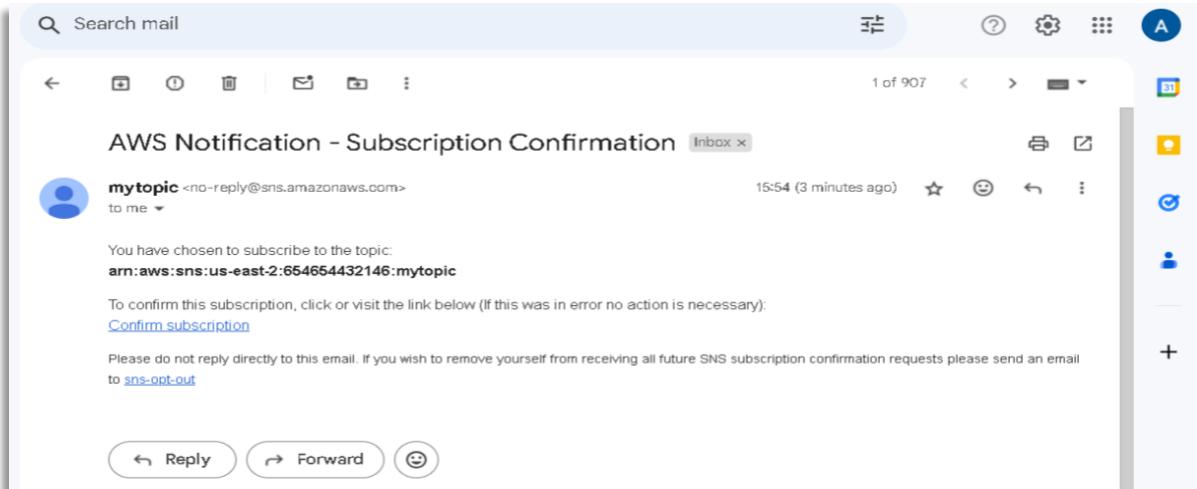
**Subscriptions**

Name mytopic	Display name mytopic
ARN arn:aws:sns:us-east-2:654654432146:mytopic	Topic owner 654654432146
Type Standard	

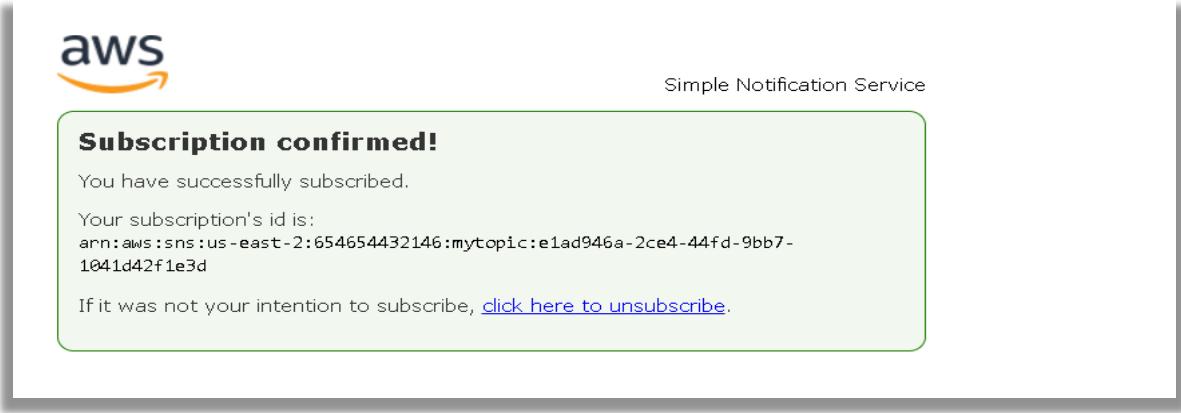
**Subscriptions (2)**

ID	Endpoint	Status	Protocol
0249ff25-f48c-4c41-9d29-813...	arn:aws:sqs:us-east-1:6546544...	Confirmed	SQS
Pending confirmation	arisia061@gmail.com	Pending confirmation	EMAIL

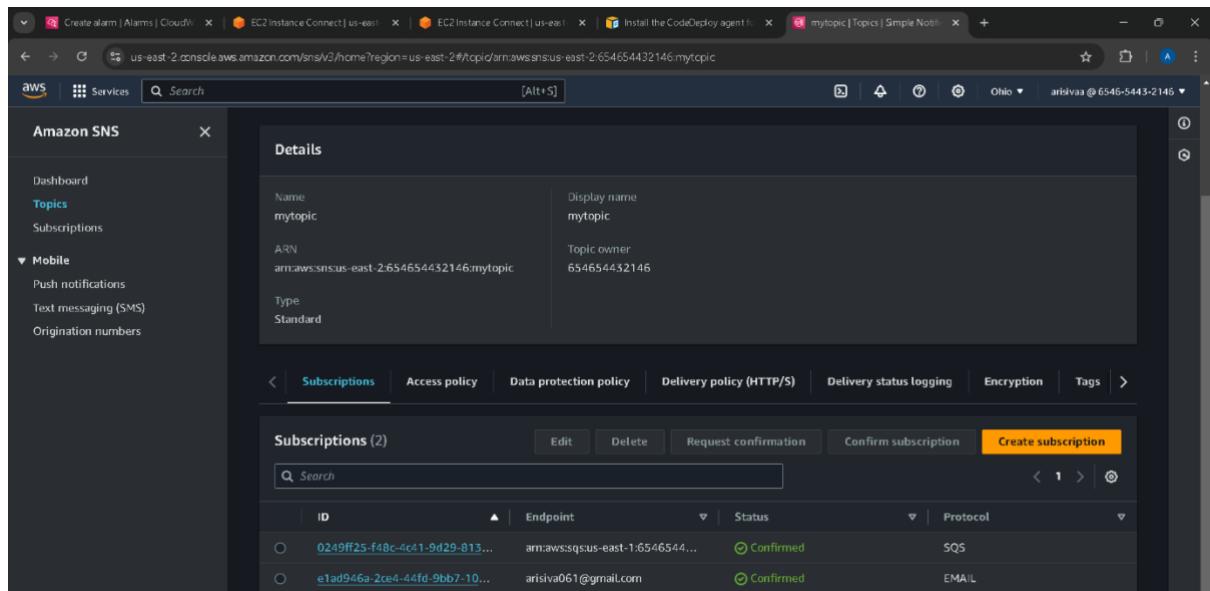
### 4) Check the Email AWS Notification -Subscription Confirmation.



## 5) Subscription confirmed.



## 6) EMAIL confirmed.



## SNS EMAIL NOTIFICATION (CLOUDWATCH)

The screenshot shows an email inbox interface with a single message highlighted. The message is from 'mytopic <no-reply@sns.amazonaws.com>' to the user. It is titled 'ALARM: "cloudwatch-200" in US East (Ohio)'. The email was received 7 minutes ago at 15:59. The message body contains a detailed description of the CloudWatch alarm state change, including the alarm name, state change reason, timestamp, AWS account, and alarm arn. It also includes a link to the AWS Management Console for further details.

ALARM: "cloudwatch-200" in US East (Ohio)

mytopic <no-reply@sns.amazonaws.com>  
to me

15:59 (7 minutes ago)

You are receiving this email because your Amazon CloudWatch Alarm "cloudwatch-200" in the US East (Ohio) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [0.43695119653288017 (30/08/24 10:19:00)] was less than the threshold (60.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Friday 30 August, 2024 10:29:05 UTC".

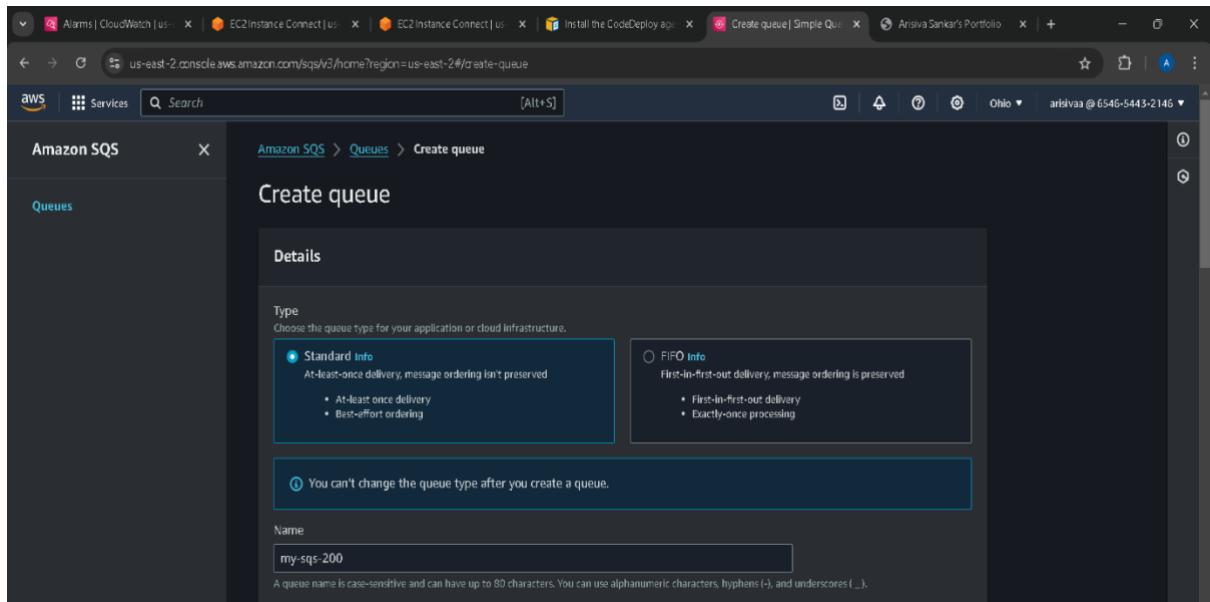
View this alarm in the AWS Management Console:  
<https://us-east-2.console.aws.amazon.com/cloudwatch/deeplink.js?region=us-east-2#alarmsV2:alarm/cloudwatch-200>

Alarm Details:

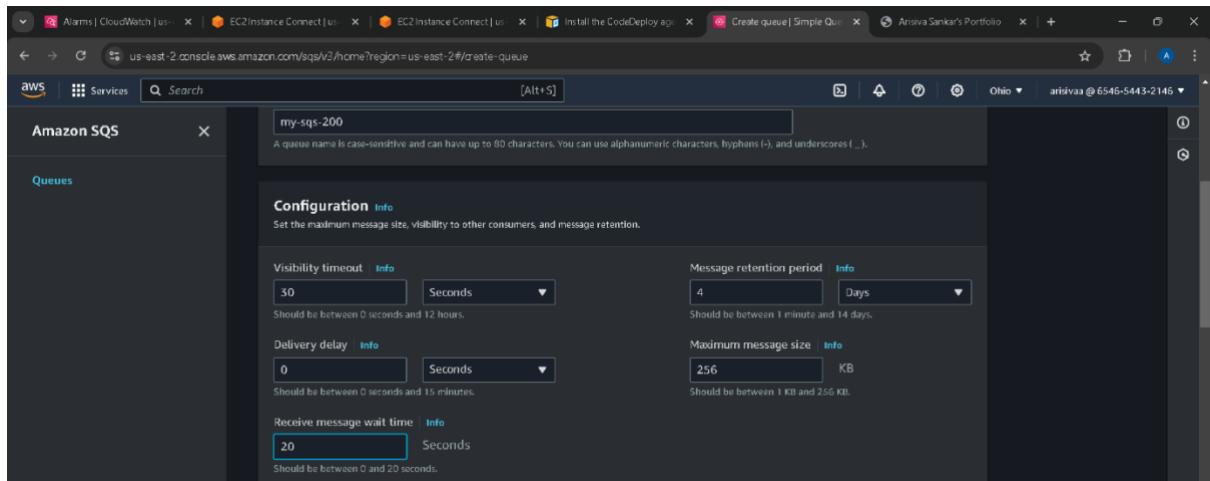
- Name: cloudwatch-200
- Description:
- State Change: INSUFFICIENT\_DATA -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [0.43695119653288017 (30/08/24 10:19:00)] was less than the threshold (60.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Friday 30 August, 2024 10:29:05 UTC
- AWS Account: 654654432146
- Alarm Arn: arn:aws:cloudwatch:us-east-2:654654432146:alarm:cloudwatch-200

## SQS ( Simple Queue Service )

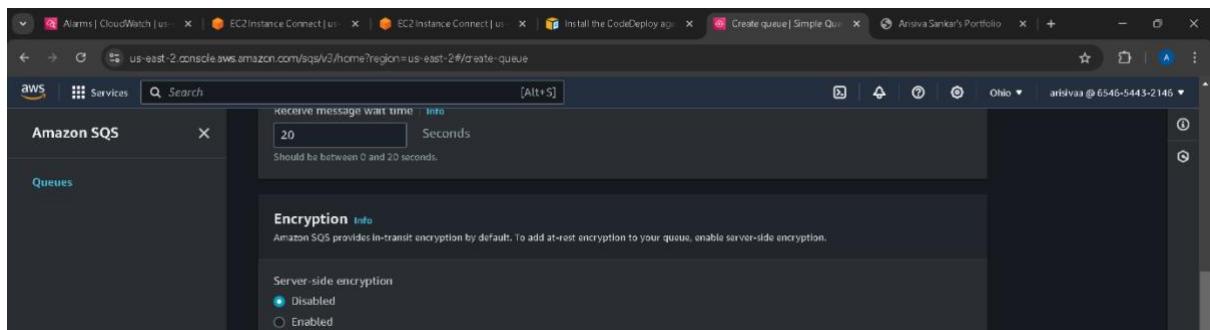
### 1) Create a queue name



## 2) Configuration time system



## 3) Encryption.



## 4) Choose a method.

The screenshot shows the 'Create queue' wizard in the AWS SQS console. The 'Choose method' step is selected, with the 'Basic' option chosen. Under 'Define who can send messages to the queue', 'Only the queue owner' is selected. Under 'Define who can receive messages from the queue', 'Only the queue owner' is selected. On the right, the JSON representation of the access policy is displayed:

```
{ "Version": "2012-10-17", "Id": "__default_policy_ID", "Statement": [{ "Sid": "__owner_statement", "Effect": "Allow", "Principal": { "AWS": "654654432146" }, "Action": ["SQS:SendMessage", "SQS:ReceiveMessage", "SQS:ListMessages"] }] }
```

## 5) Successfully SQS created.

The screenshot shows the 'Queue details' page for the 'my-sqs-200' queue. A green banner at the top indicates 'Subscribed successfully to topic arn:aws:sns:us-east-2:654654432146:mytopic.' Below the banner, the queue name is 'my-sqs-200'. The 'Details' tab is selected, showing the following information:

Name	Type	ARN
my-sqs-200	Standard	arn:aws:sqs:us-east-2:654654432146:my-sqs-200
Encryption	Disabled	URL https://sqs.us-east-2.amazonaws.com/654654432146/my-sqs-200
		Dead-letter queue -

## 6) SNS Subscription.

The screenshot shows the 'SNS subscriptions' tab for the 'my-sqs-200' queue. It lists one subscription:

SNS subscriptions (1)	Subscription ARN	Topic ARN
arn:aws:sns:us-east-2:654654432146:mytopic:9a058bad-fdce-4ba8-89c6-294b5c5ac6a1	arn:aws:sns:us-east-2:654654432146:mytopic	

## 7) SQS Notification.

