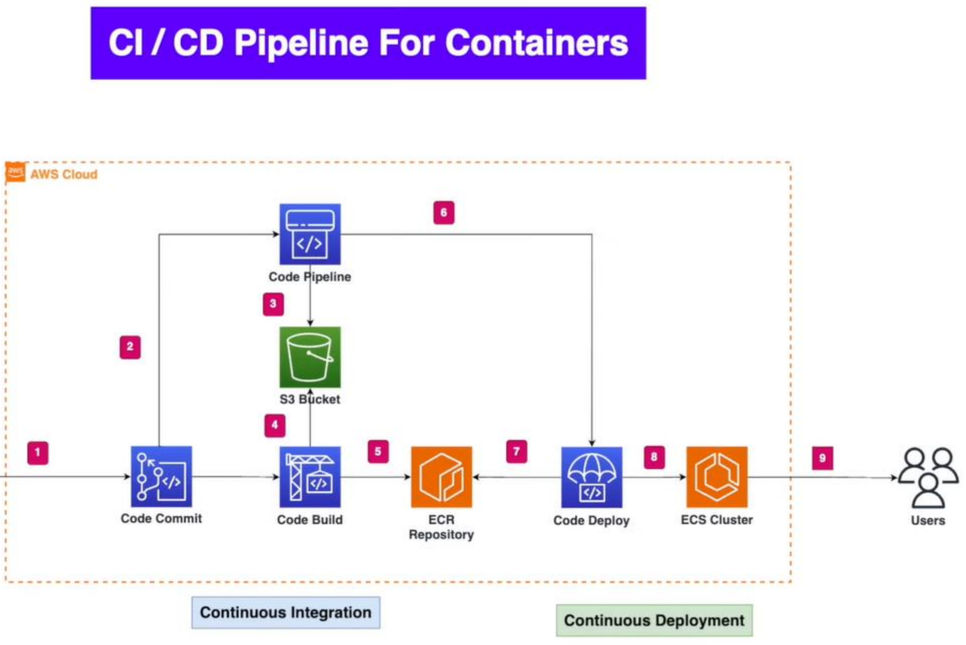
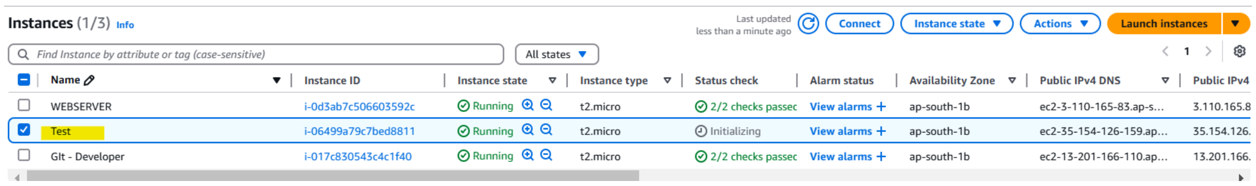
AWS Project - Deploy Docker Container to AWS ECS Automatically with CI CD Pipeline | Step by Step



Step 1 : Launch & Deploy WebApp Image to AWS ECS Fargate.

Step 2 : CI/CD Pipeline for AWS ECS using: CodeCommit, CodeBuild and CodePipeline.

Create a New instance.

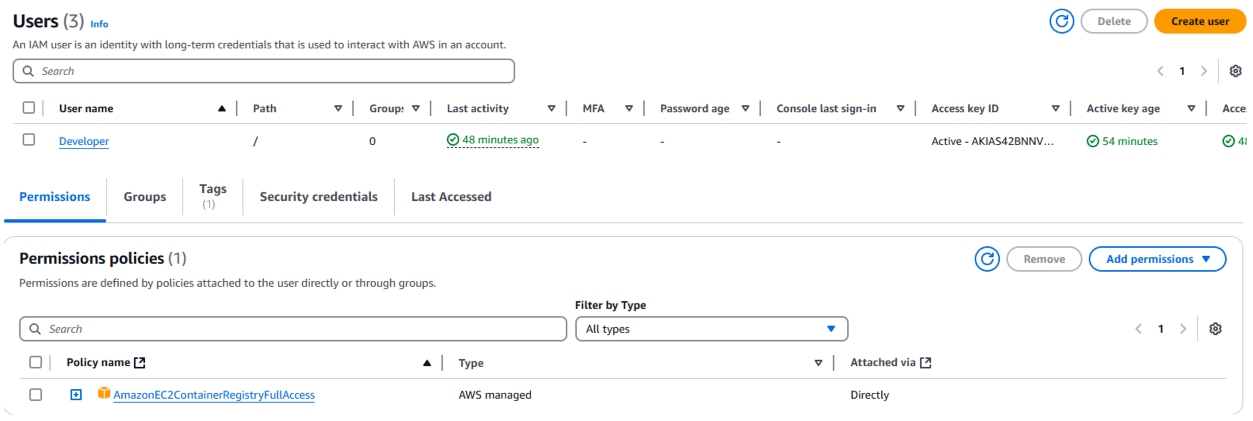

Instances (1/3) Info
Last updated less than a minute ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
1

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
☐	WEBSERVER	i-0d3ab7c506603592c	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1b	ec2-3-110-165-83.ap-s...	3.110.165.8
☑	Test	i-06499a79c7bed8811	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-35-154-126-159.ap...	35.154.126.
☐	GIt - Developer	i-017c830543c4c1f40	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1b	ec2-13-201-166-110.ap...	13.201.166.

<u>Using instance is Ubuntu 22.04</u>

Create AWS-CLI user:

Create IAM user and attached the Policies and Create the Access key and Secrete Key in developer user.



Users (3) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active key age	Acce
Developer	/	0	48 minutes ago	-	-	-	Active - AKIAS42BNNV...	54 minutes	4

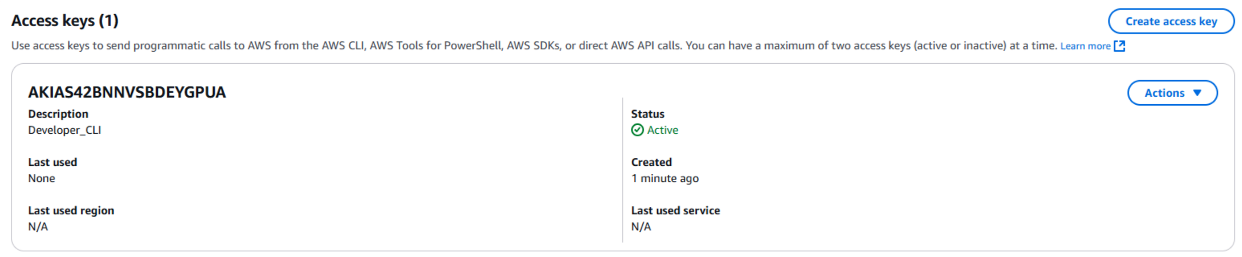
Permissions | Groups | Tags (1) | Security credentials | Last Accessed

Permissions policies (1)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type: All types

Policy name	Type	Attached via
AmazonEC2ContainerRegistryFullAccess	AWS managed	Directly

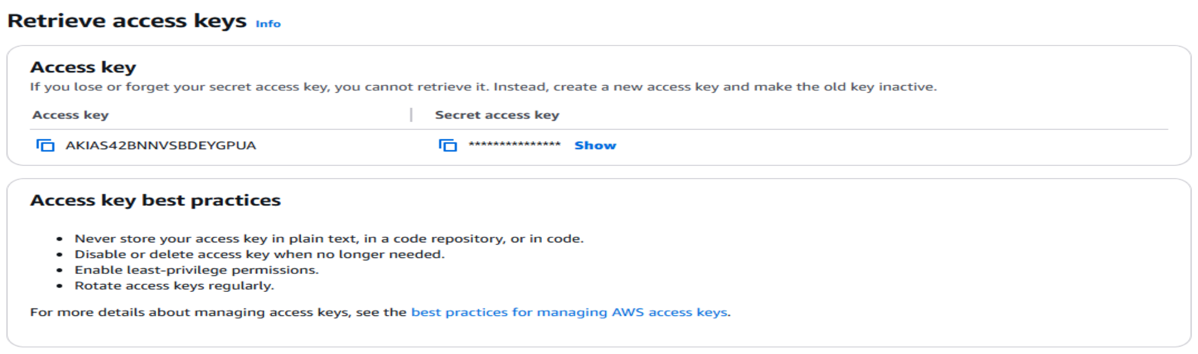


Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

AKIAS42BNNVSBDEYGPUA

Description Developer_CLI	**Status** Active
Last used None	**Created** 1 minute ago
Last used region N/A	**Last used service** N/A



Retrieve access keys Info

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIAS42BNNVSBDEYGPUA	*************** Show

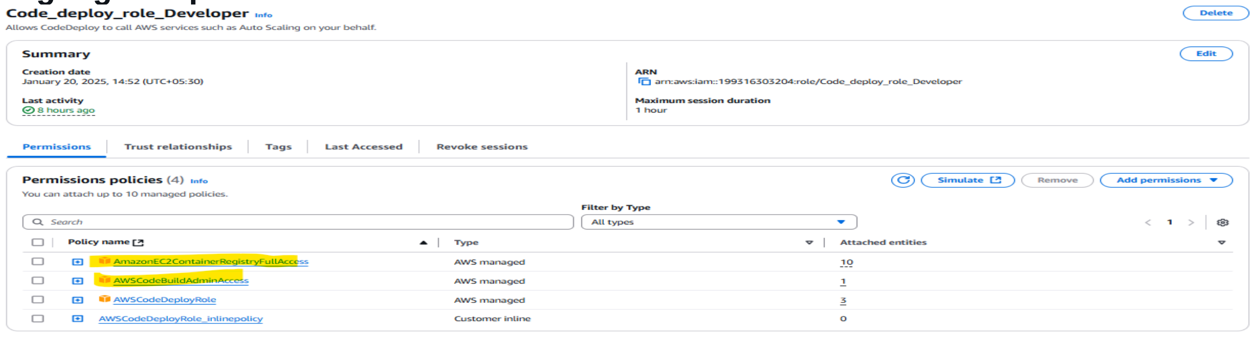
Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.

Create Role Code_deploy_role_Developer.

Highlighted permission is must.



Code_deploy_role_Developer Info

Allows CodeDeploy to call AWS services such as Auto Scaling on your behalf.

Summary

Creation date January 20, 2025, 14:52 (UTC+05:30)	**ARN** arn:aws:iam::199316303204:role/Code_deploy_role_Developer
Last activity 8 hours ago	**Maximum session duration** 1 hour

Permissions | Trust relationships | Tags | Last Accessed | Revoke sessions

Permissions policies (4) Info

You can attach up to 10 managed policies.

Filter by Type: All types

Policy name	Type	Attached entities
AmazonEC2ContainerRegistryFullAccess	AWS managed	10
AWSCodeBuildAdminAccess	AWS managed	1
AWSCodeDeployRole	AWS managed	3
AWSCodeDeployRole_inlinepolicy	Customer inline	0

Login the instance via ssh and configure the awscli.

```
sudo apt -y update

sudo apt -y upgrade

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
"awscliv2.zip"

apt install unzip

unzip awscliv2.zip

sudo ./aws/install

aws configure
```

After That Install the Docker:

docker installation in ubuntu 22.04

```
sudo apt-get update

sudo apt-get install ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
/etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

echo   "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" |   sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin

sudo systemctl status docker
```

GIT Install

sudo apt install git

git -version

Nodejs Install

sudo apt update

sudo apt install nodejs

node -v

sudo apt install npm

cd ~

curl -sL https://deb.nodesource.com/setup_20.x -o nodesource_setup.sh

nano nodesource_setup.sh

sudo bash nodesource_setup.sh

sudo apt install nodejs

node -v

Installing Node Using the Node Version Manager

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.1/install.sh

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.1/install.sh | bash

source ~/.bashrc

nvm list-remote

nvm install v20.18.0

nvm list

nvm install lts/fermium

node -v

If you want to Removing Node.js

sudo apt remove nodejs

sudo apt purge nodejs

nvm current

nvm uninstall node_version

nvm deactivate

Build Web APP Image

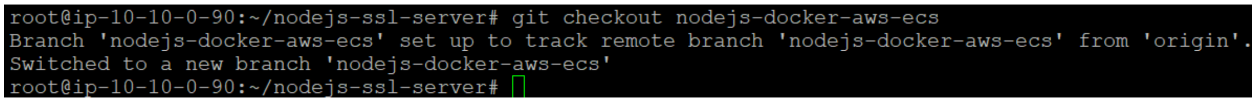
Git repo clone first in Test server.

git clone https://github.com/saasscaleup/nodejs-ssl-server.git

After that the.

cd nodejs-ssl-server/

git checkout nodejs-docker-aws-ecs

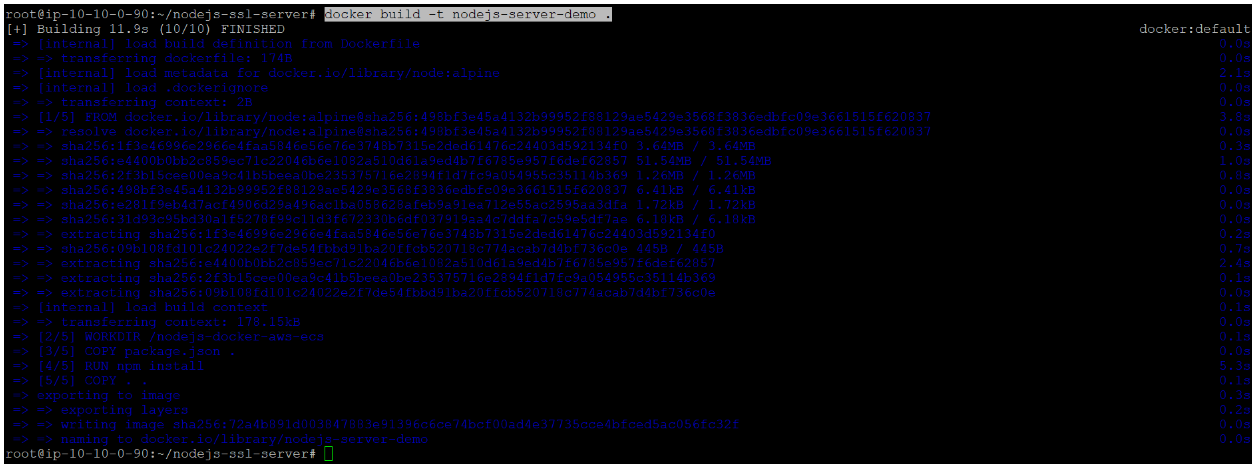


```
root@ip-10-10-0-90:~/nodejs-ssl-server# git checkout nodejs-docker-aws-ecs
Branch 'nodejs-docker-aws-ecs' set up to track remote branch 'nodejs-docker-aws-ecs' from 'origin'.
Switched to a new branch 'nodejs-docker-aws-ecs'
root@ip-10-10-0-90:~/nodejs-ssl-server# 
```

ls -la

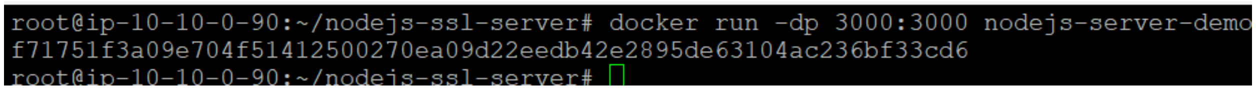
docker images -a

docker build -t nodejs-server-demo .



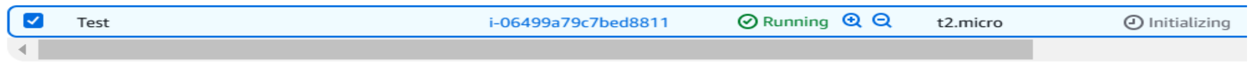
```
root@ip-10-10-0-90:~/nodejs-ssl-server# docker build -t nodejs-server-demo .
[+] Building 11.9s (10/10) FINISHED                                                                         docker:default
 => [internal] load build definition from Dockerfile                                                                   0.0s
 => => transferring dockerfile: 174B                                                                                   0.0s
 => [internal] load metadata for docker.io/library/node:alpine                                                         2.1s
 => [internal] load .dockerignore                                                                                      0.0s
 => => transferring context: 2B                                                                                        0.0s
 => [1/5] FROM docker.io/library/node:alpine@sha256:498bf3e45a4132b99952f88129ae5429e3568f3836edbfc09e3661515f620837   3.8s
 => => resolve docker.io/library/node:alpine@sha256:498bf3e45a4132b99952f88129ae5429e3568f3836edbfc09e3661515f620837   0.0s
 => => sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0 3.64MB / 3.64MB                         0.3s
 => => sha256:e4400b0bb2c859ec71c22046b6e1082a510d61a9ed4b7f6785e957f6def62857 51.54MB / 51.54MB                       1.0s
 => => sha256:2f3b15cee00ea9c41b5beea0be235375716e2894f1d7fc9a054955c35114b369 1.26MB / 1.26MB                         0.8s
 => => sha256:498bf3e45a4132b99952f88129ae5429e3568f3836edbfc09e3661515f620837 6.41kB / 6.41kB                         0.0s
 => => sha256:e281f9eb4d7acf4906d29a496ac1ba058628afeb9a91ea712e55ac2595aa3dfa 1.72kB / 1.72kB                         0.0s
 => => sha256:31d93c95bd30a1f5278f99c11d3f672330b6df037919aa4c7ddfa7c59e5df7ae 6.18kB / 6.18kB                         0.0s
 => => extracting sha256:1f3e46996e2966e4faa5846e56e76e3748b7315e2ded61476c24403d592134f0                              0.2s
 => => sha256:09b108fd101c24022e2f7de54fbbd91ba20ffcb520718c774acab7d4bf736c0e 445B / 445B                             0.7s
 => => extracting sha256:e4400b0bb2c859ec71c22046b6e1082a510d61a9ed4b7f6785e957f6def62857                              2.4s
 => => extracting sha256:2f3b15cee00ea9c41b5beea0be235375716e2894f1d7fc9a054955c35114b369                              0.1s
 => => extracting sha256:09b108fd101c24022e2f7de54fbbd91ba20ffcb520718c774acab7d4bf736c0e                              0.0s
 => [internal] load build context                                                                                      0.1s
 => => transferring context: 178.15kB                                                                                  0.0s
 => [2/5] WORKDIR /nodejs-docker-aws-ecs                                                                               0.1s
 => [3/5] COPY package.json .                                                                                          0.0s
 => [4/5] RUN npm install                                                                                              5.3s
 => [5/5] COPY . .                                                                                                     0.1s
 => exporting to image                                                                                                 0.3s
 => => exporting layers                                                                                                0.2s
 => => writing image sha256:72a4b891d003847883e91396c6ce74bcf00ad4e37735cce4bfced5ac056fc32f                           0.0s
 => => naming to docker.io/library/nodejs-server-demo                                                                  0.0s
root@ip-10-10-0-90:~/nodejs-ssl-server# 
```

docker run -dp 3000:3000 nodejs-server-demo

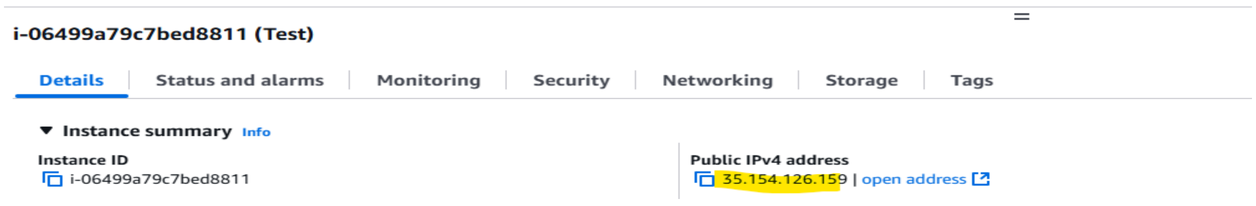


```
root@ip-10-10-0-90:~/nodejs-ssl-server# docker run -dp 3000:3000 nodejs-server-demo
f71751f3a09e704f51412500270ea09d22eedb42e2895de63104ac236bf33cd6
root@ip-10-10-0-90:~/nodejs-ssl-server# 
```

After that go to aws console Test server public IP copy to put the website using port 3000.



	Test	i-06499a79c7bed8811	Running	t2.micro	Initializing



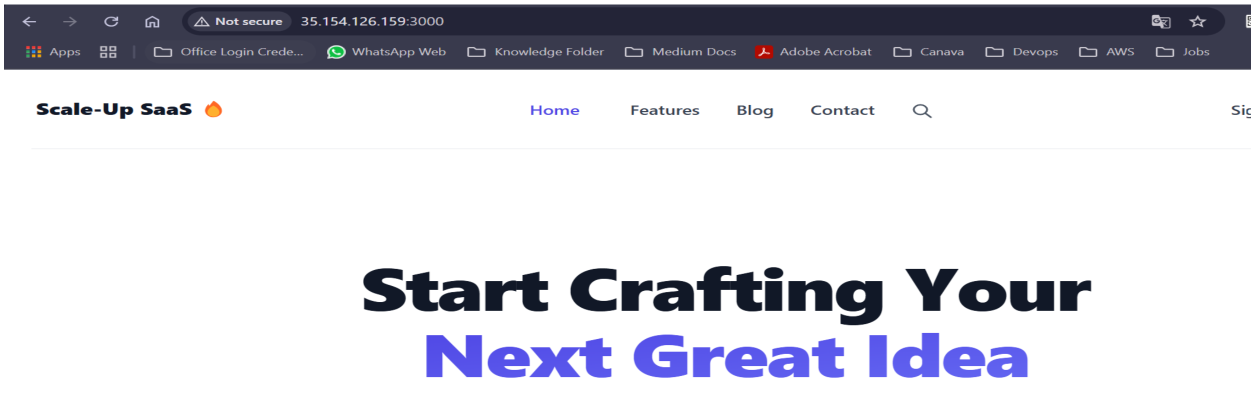
i-06499a79c7bed8811 (Test)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance summary Info

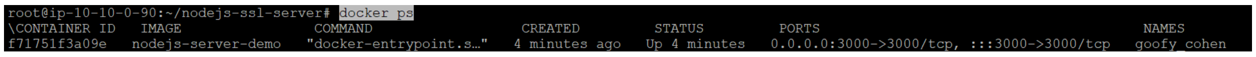
Instance ID
i-06499a79c7bed8811

Public IPv4 address
35.154.126.159 | open address



After go to the ssh.

docker ps

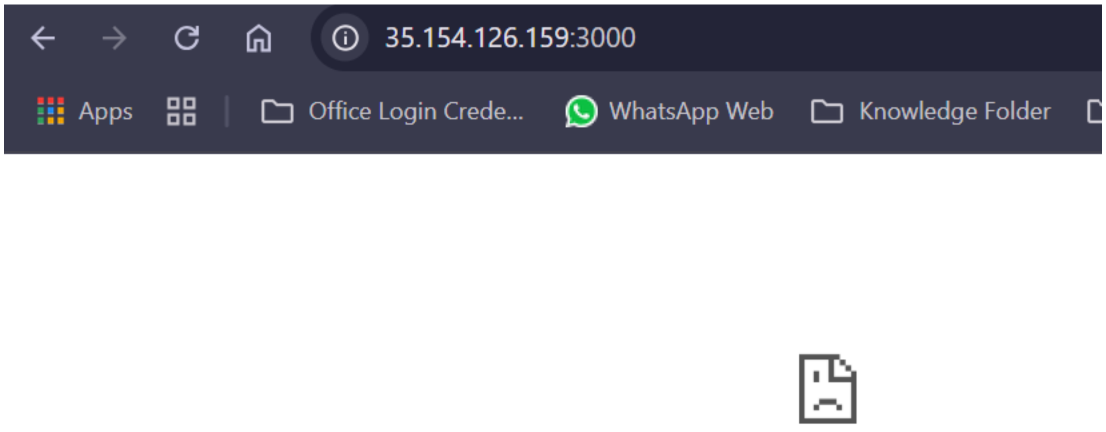


```
root@ip-10-10-0-90:~/nodejs-ssl-server# docker ps
\CONTAINER ID   IMAGE                COMMAND                  CREATED         STATUS         PORTS                                       NAMES
f71751f3a09e   nodejs-server-demo   "docker-entrypoint.s…"   4 minutes ago   Up 4 minutes   0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   goofy_cohen
```

Remove the Image.

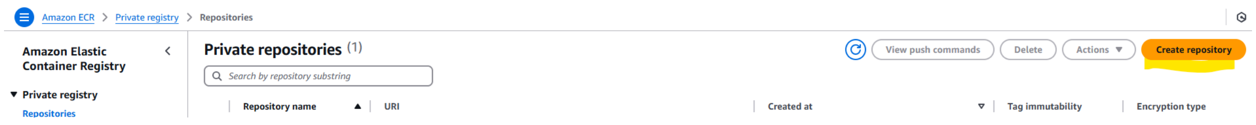
docker rm -f f71751f3a09e

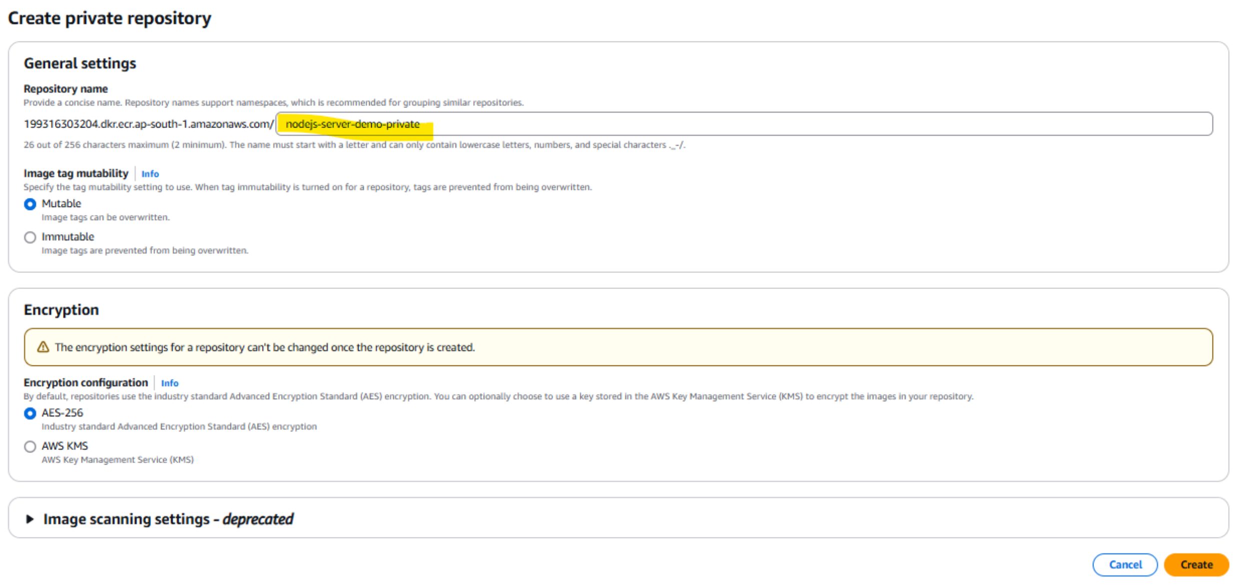
After Check the site is not working.



Create and Push Image to AWS ECR Repository:

Open the ECR service create the repository in Private repository.





Create private repository

General settings

Repository name

Provide a concise name. Repository names support namespaces, which is recommended for grouping similar repositories.

199316303204.dkr.ecr.ap-south-1.amazonaws.com/ nodejs-server-demo-private

26 out of 256 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, and special characters ._-/.

Image tag mutability | Info

Specify the tag mutability setting to use. When tag immutability is turned on for a repository, tags are prevented from being overwritten.

[✓ Mutable] Image tags can be overwritten.

Immutable Image tags are prevented from being overwritten.

Encryption

The encryption settings for a repository can't be changed once the repository is created.

Encryption configuration | Info

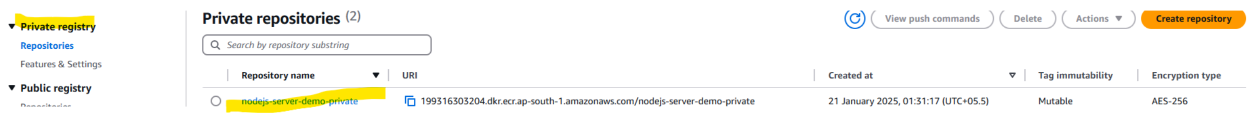
By default, repositories use the industry standard Advanced Encryption Standard (AES) encryption. You can optionally choose to use a key stored in the AWS Key Management Service (KMS) to encrypt the images in your repository.

[✓ AES-256] Industry standard Advanced Encryption Standard (AES) encryption

AWS KMS AWS Key Management Service (KMS)

Image scanning settings - *deprecated*

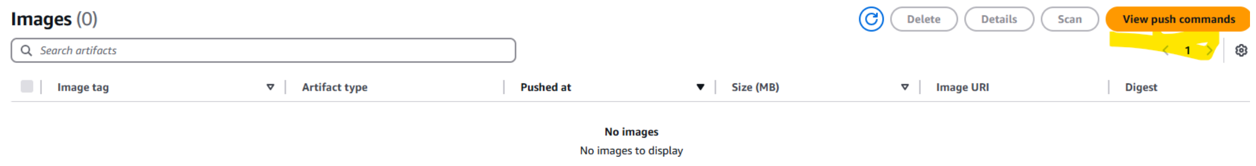
Cancel Create



Private repositories (2)

Repository name	URI	Created at	Tag immutability	Encryption type
nodejs-server-demo-private	199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private	21 January 2025, 01:31:17 (UTC+05.5)	Mutable	AES-256

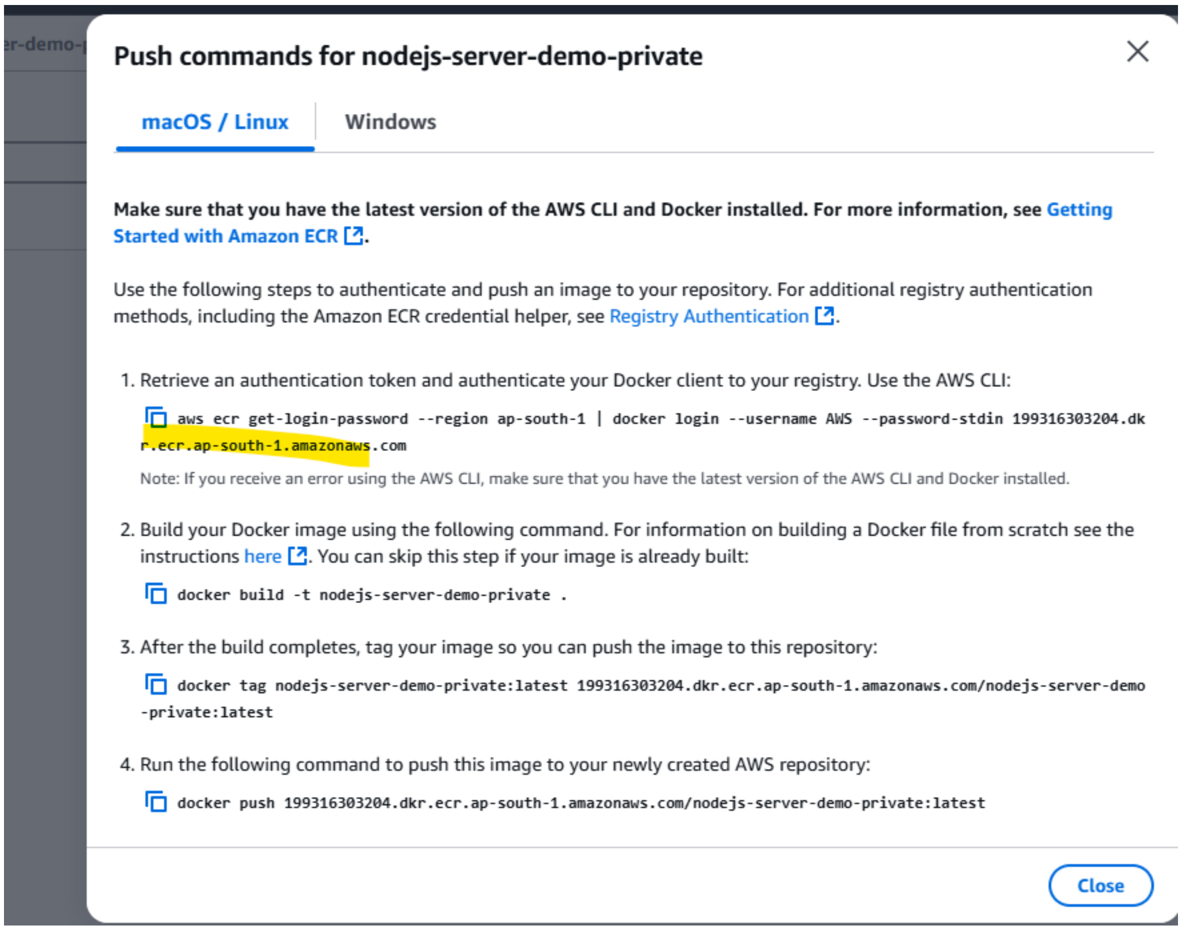
Go to the repository and click the push command and following the commands one by one.



Images (0)

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest

No images
No images to display



Push commands for nodejs-server-demo-private

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see Getting Started with Amazon ECR.

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see Registry Authentication.

1. Retrieve an authentication token and authenticate your Docker client to your registry. Use the AWS CLI:

```
aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 199316303204.dkr.ecr.ap-south-1.amazonaws.com
```

Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.

2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:

```
docker build -t nodejs-server-demo-private .
```

3. After the build completes, tag your image so you can push the image to this repository:

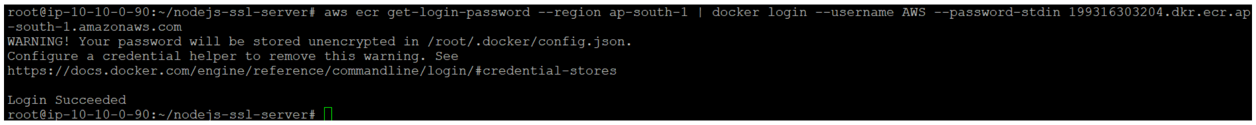
```
docker tag nodejs-server-demo-private:latest 199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private:latest
```

4. Run the following command to push this image to your newly created AWS repository:

```
docker push 199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private:latest
```

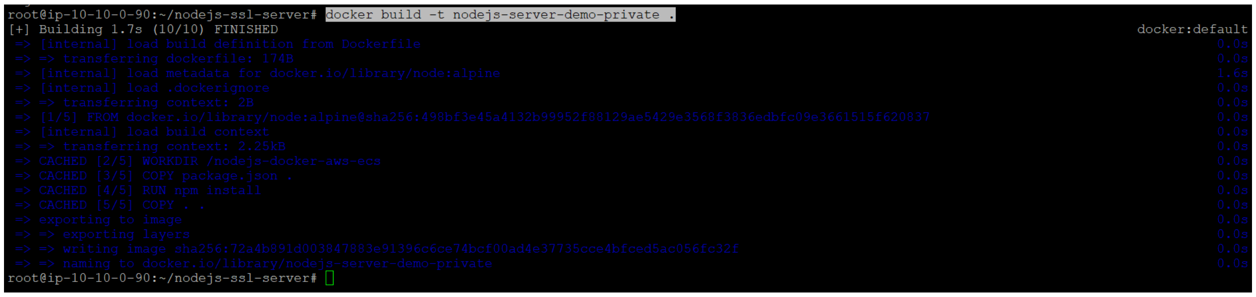
Copy the command to paste the ssh in test server.

aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 199316303204.dkr.ecr.ap-south-1.amazonaws.com



```
root@ip-10-10-0-90:~/nodejs-ssl-server# aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 199316303204.dkr.ecr.ap
-south-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
root@ip-10-10-0-90:~/nodejs-ssl-server#
```

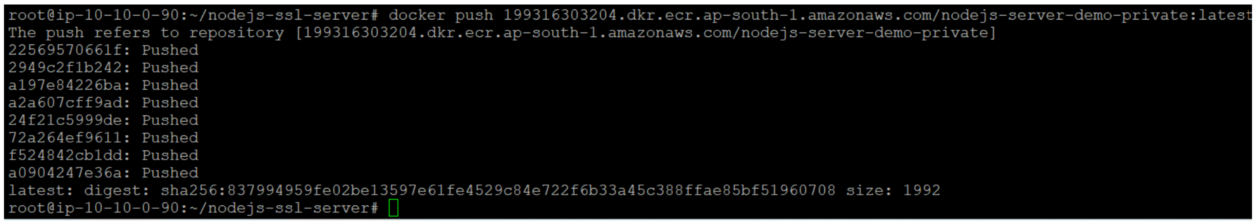
docker build -t nodejs-server-demo-private .



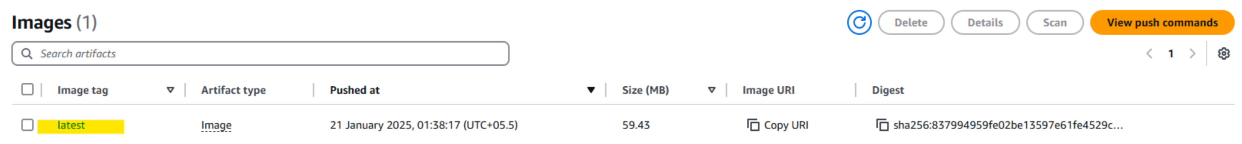
```
root@ip-10-10-0-90:~/nodejs-ssl-server# docker build -t nodejs-server-demo-private .
[+] Building 1.7s (10/10) FINISHED                                                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                                                                   0.0s
 => => transferring dockerfile: 174B                                                                                                                   0.0s
 => [internal] load metadata for docker.io/library/node:alpine                                                                                         1.6s
 => [internal] load .dockerignore                                                                                                                      0.0s
 => => transferring context: 2B                                                                                                                        0.0s
 => [1/5] FROM docker.io/library/node:alpine@sha256:498bf3e45a4132b99952f88129ae5429e3568f3836edbfc09e3661515f620837                                   0.0s
 => [internal] load build context                                                                                                                      0.0s
 => => transferring context: 2.25kB                                                                                                                    0.0s
 => CACHED [2/5] WORKDIR /nodejs-docker-aws-ecs                                                                                                        0.0s
 => CACHED [3/5] COPY package.json .                                                                                                                   0.0s
 => CACHED [4/5] RUN npm install                                                                                                                       0.0s
 => CACHED [5/5] COPY . .                                                                                                                              0.0s
 => exporting to image                                                                                                                                 0.0s
 => => exporting layers                                                                                                                                0.0s
 => => writing image sha256:72a4b891d003847883e91396c6ce74bcf00ad4e37735cce4bfced5ac056fc32f                                                           0.0s
 => => naming to docker.io/library/nodejs-server-demo-private                                                                                          0.0s
root@ip-10-10-0-90:~/nodejs-ssl-server#
```

docker tag nodejs-server-demo-private:latest 199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private:latest

docker push 199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private:latest



```
root@ip-10-10-0-90:~/nodejs-ssl-server# docker push 199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private:latest
The push refers to repository [199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private]
22569570661f: Pushed
2949c2f1b242: Pushed
a197e84226ba: Pushed
a2a607cff9ad: Pushed
24f21c5999de: Pushed
72a264ef9611: Pushed
f524842cb1dd: Pushed
a0904247e36a: Pushed
latest: digest: sha256:837994959fe02be13597e61fe4529c84e722f6b33a45c388ffae85bf51960708 size: 1992
root@ip-10-10-0-90:~/nodejs-ssl-server#
```

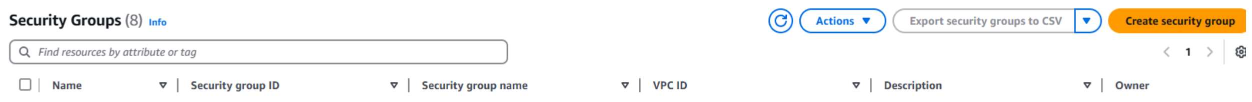


Images (1)

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest
latest	Image	21 January 2025, 01:38:17 (UTC+05.5)	59.43	Copy URI	sha256:837994959fe02be13597e61fe4529c...

Create Security Groups:

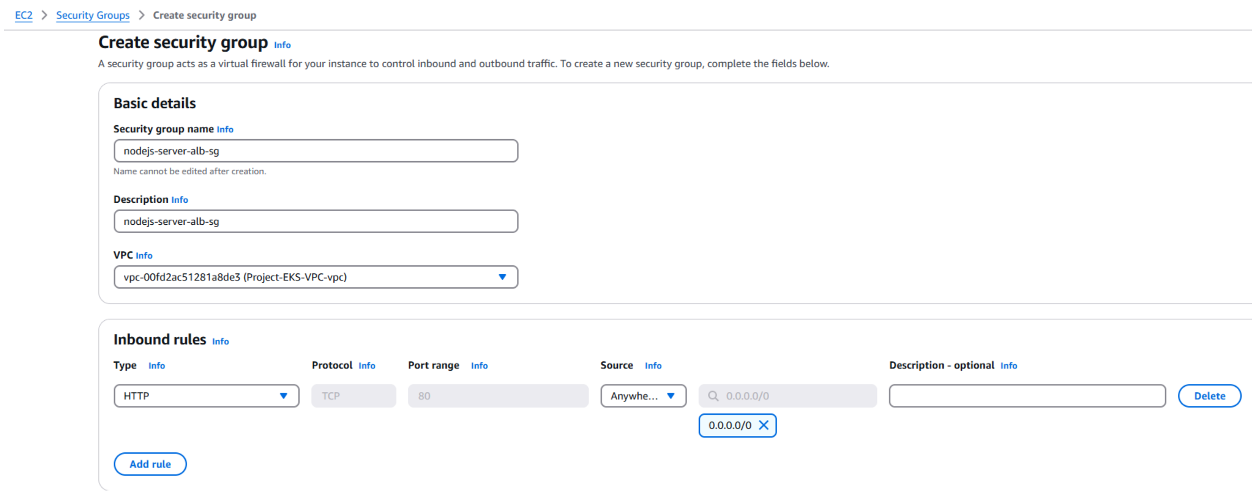
Create 2 security groups.



Security Groups (8) Info

Actions
Export security groups to CSV
Create security group
Find resources by attribute or tag

Name	Security group ID	Security group name	VPC ID	Description	Owner



EC2 > Security Groups > Create security group

Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name: nodejs-server-alb-sg

Name cannot be edited after creation.

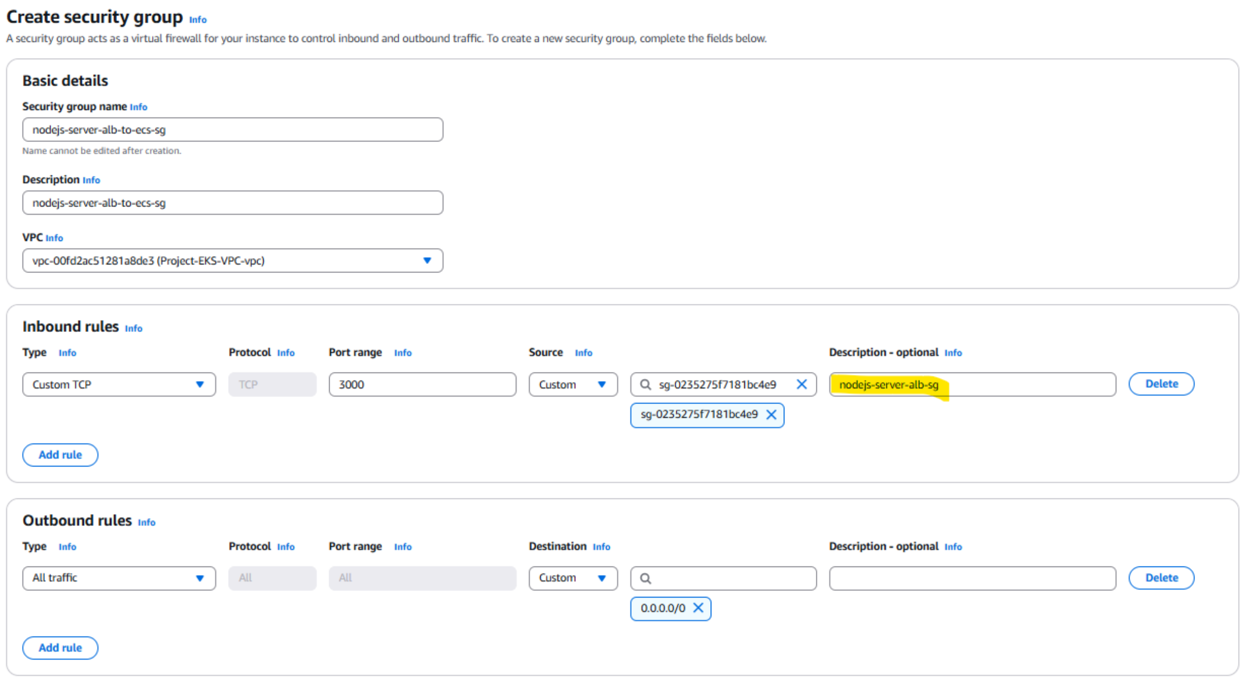
Description: nodejs-server-alb-sg

VPC: vpc-00fd2ac51281a8de3 (Project-EKS-VPC-vpc)

Inbound rules

Type	Protocol	Port range	Source		Description - optional
HTTP	TCP	80	Anywhe...	0.0.0.0/0	

Inbound rules select nodejs-server-alb-sg.



Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name: nodejs-server-alb-to-ecs-sg

Name cannot be edited after creation.

Description: nodejs-server-alb-to-ecs-sg

VPC: vpc-00fd2ac51281a8de3 (Project-EKS-VPC-vpc)

Inbound rules

Type	Protocol	Port range	Source		Description - optional
Custom TCP	TCP	3000	Custom	sg-0235275f7181bc4e9	nodejs-server-alb-sg

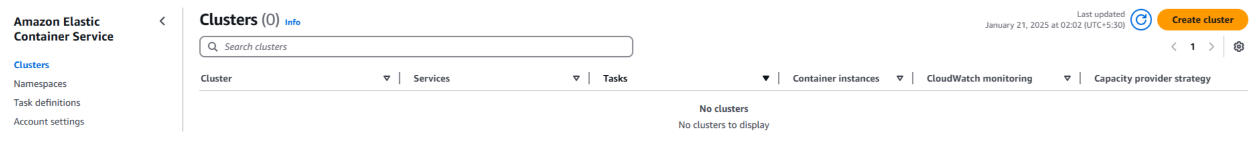
Outbound rules

Type	Protocol	Port range	Destination		Description - optional
All traffic	All	All	Custom	0.0.0.0/0	

And create it.

Create ECS Fargate Cluster:

Open the ECS service.



Amazon Elastic Container Service

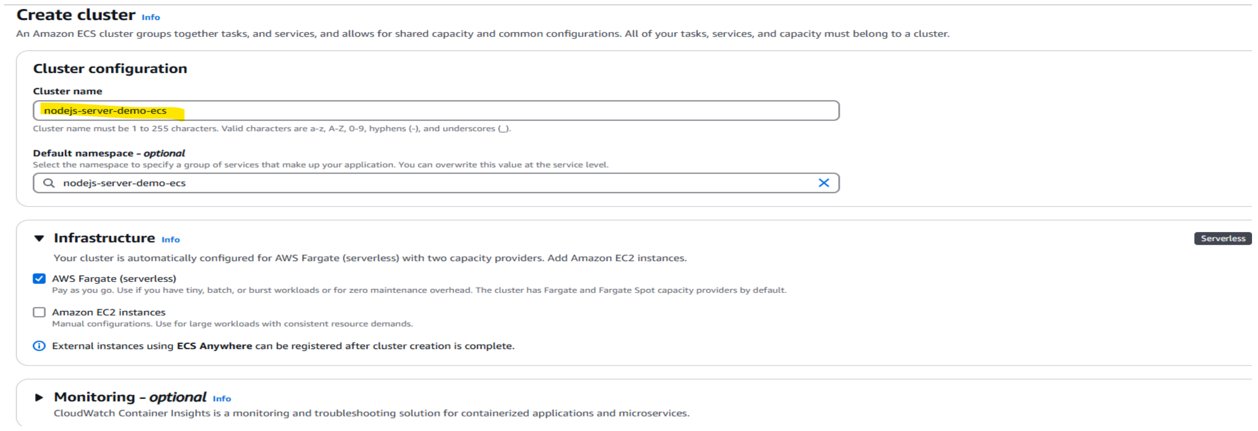
Clusters
Namespaces
Task definitions
Account settings

Clusters (0) Info

Last updated January 21, 2025 at 02:02 (UTC+5:30)

Cluster	Services	Tasks	Container instances	CloudWatch monitoring	Capacity provider strategy

No clusters
No clusters to display



Create cluster Info

An Amazon ECS cluster groups together tasks, and services, and allows for shared capacity and common configurations. All of your tasks, services, and capacity must belong to a cluster.

Cluster configuration

Cluster name: nodejs-server-demo-ecs
Cluster name must be 1 to 255 characters. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).

Default namespace - *optional*
Select the namespace to specify a group of services that make up your application. You can overwrite this value at the service level.
nodejs-server-demo-ecs

Infrastructure Info — Serverless

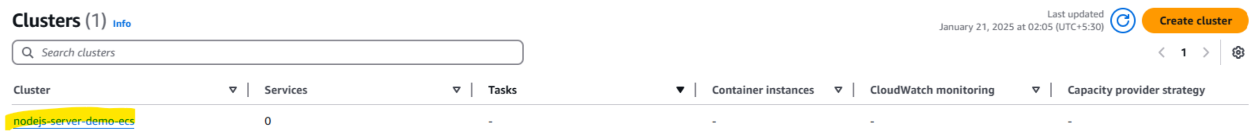
Your cluster is automatically configured for AWS Fargate (serverless) with two capacity providers. Add Amazon EC2 instances.

- [x] AWS Fargate (serverless)
  Pay as you go. Use if you have tiny, batch, or burst workloads or for zero maintenance overhead. The cluster has Fargate and Fargate Spot capacity providers by default.
- [ ] Amazon EC2 instances
  Manual configurations. Use for large workloads with consistent resource demands.

External instances using **ECS Anywhere** can be registered after cluster creation is complete.

Monitoring - *optional* Info
CloudWatch Container Insights is a monitoring and troubleshooting solution for containerized applications and microservices.

Create it.

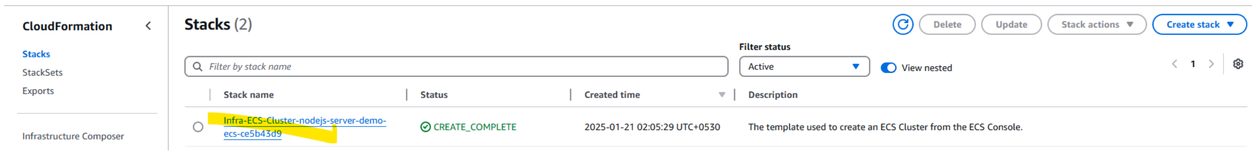


Clusters (1) Info

Last updated January 21, 2025 at 02:05 (UTC+5:30)

Cluster	Services	Tasks	Container instances	CloudWatch monitoring	Capacity provider strategy
nodejs-server-demo-ecs	0	-	-	-	-

Open the service Cloud Formation and already stack has been created.



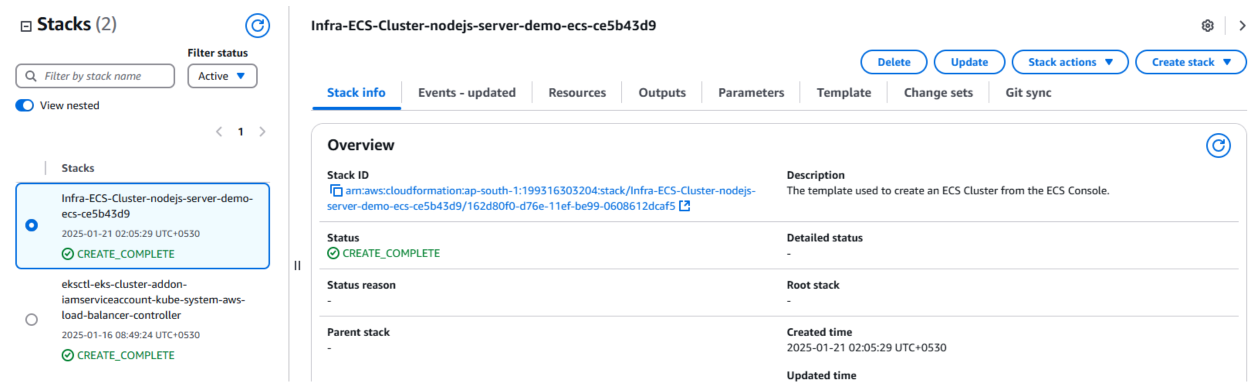
CloudFormation

Stacks
StackSets
Exports
Infrastructure Composer

Stacks (2)

Filter status: Active — View nested

Stack name	Status	Created time	Description
Infra-ECS-Cluster-nodejs-server-demo-ecs-ce5b43d9	CREATE_COMPLETE	2025-01-21 02:05:29 UTC+0530	The template used to create an ECS Cluster from the ECS Console.



Stacks (2)

Filter status: Active — View nested

Stacks
Infra-ECS-Cluster-nodejs-server-demo-ecs-ce5b43d9 2025-01-21 02:05:29 UTC+0530 CREATE_COMPLETE
eksctl-eks-cluster-addon-iamserviceaccount-kube-system-aws-load-balancer-controller 2025-01-16 08:49:24 UTC+0530 CREATE_COMPLETE

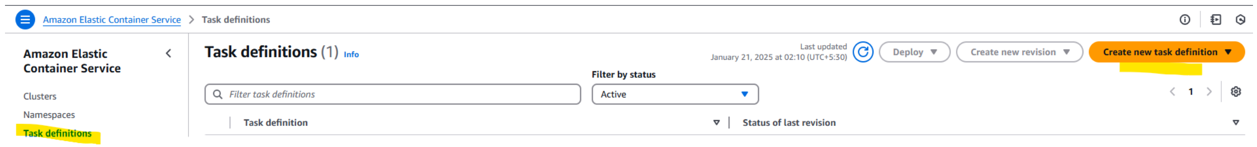
Infra-ECS-Cluster-nodejs-server-demo-ecs-ce5b43d9

Stack info | Events - updated | Resources | Outputs | Parameters | Template | Change sets | Git sync

Overview

Stack ID: arn:aws:cloudformation:ap-south-1:199316303204:stack/Infra-ECS-Cluster-nodejs-server-demo-ecs-ce5b43d9/162d80f0-d76e-11ef-be99-0608612dcaf5	Description: The template used to create an ECS Cluster from the ECS Console.
Status: CREATE_COMPLETE	Detailed status: -
Status reason: -	Root stack: -
Parent stack: -	Created time: 2025-01-21 02:05:29 UTC+0530
	Updated time

Create Task Definition:


Amazon Elastic Container Service > Task definitions

Amazon Elastic Container Service
- Clusters
- Namespaces
- Task definitions

Task definitions (1) Info

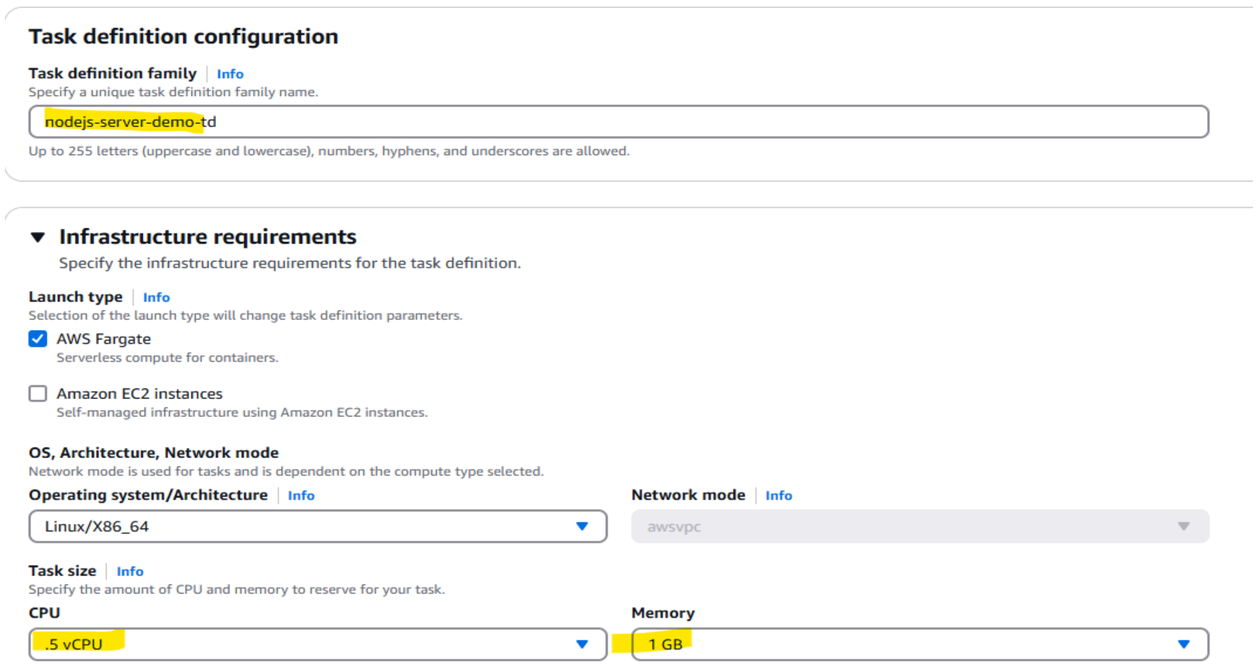
Last updated January 21, 2025 at 02:10 (UTC+5:30)

Deploy | Create new revision | Create new task definition

Filter task definitions

Filter by status: Active

Task definition	Status of last revision


Task definition configuration

Task definition family Info
Specify a unique task definition family name.
nodejs-server-demo-td
Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.

Infrastructure requirements
Specify the infrastructure requirements for the task definition.

Launch type Info
Selection of the launch type will change task definition parameters.
- [x] AWS Fargate
  Serverless compute for containers.
- [ ] Amazon EC2 instances
  Self-managed infrastructure using Amazon EC2 instances.

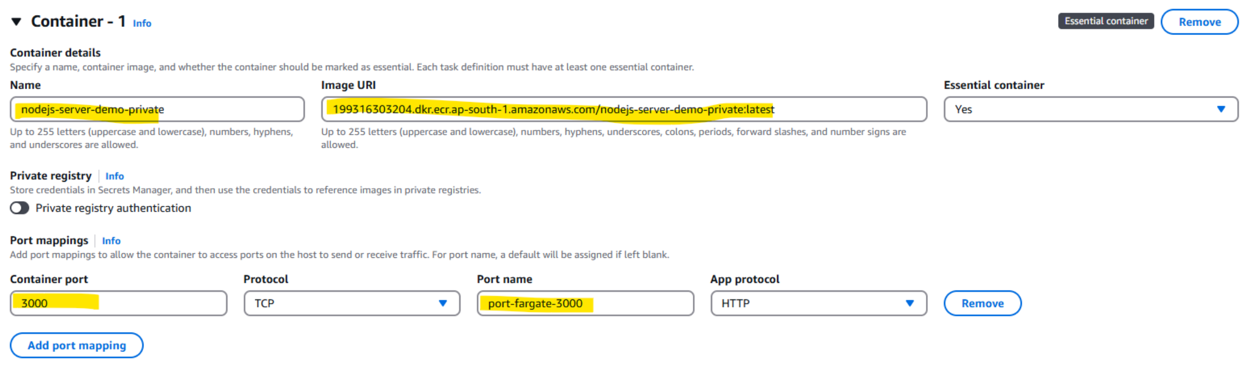
OS, Architecture, Network mode
Network mode is used for tasks and is dependent on the compute type selected.

Operating system/Architecture	Network mode
Linux/X86_64	awsvpc

Task size Info
Specify the amount of CPU and memory to reserve for your task.

CPU	Memory
.5 vCPU	1 GB

Container details get into ECR service and Image uri last added the tag **latest** and port 3000 change it.


Container - 1 Info — Essential container | Remove

Container details
Specify a name, container image, and whether the container should be marked as essential. Each task definition must have at least one essential container.

Name	Image URI	Essential container
nodejs-server-demo-private	199316303204.dkr.ecr.ap-south-1.amazonaws.com/nodejs-server-demo-private:latest	Yes

Up to 255 letters (uppercase and lowercase), numbers, hyphens, and underscores are allowed.
Up to 255 letters (uppercase and lowercase), numbers, hyphens, underscores, colons, periods, forward slashes, and number signs are allowed.

Private registry Info
Store credentials in Secrets Manager, and then use the credentials to reference images in private registries.
Private registry authentication

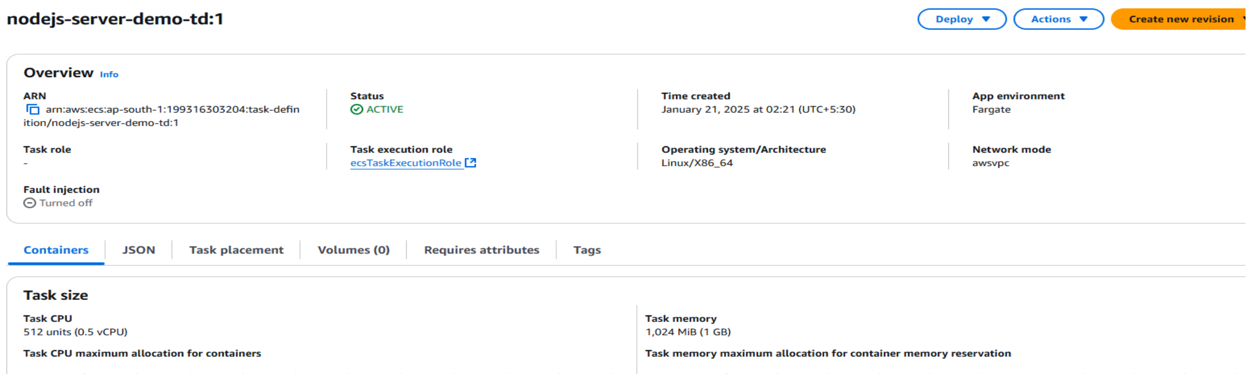
Port mappings Info
Add port mappings to allow the container to access ports on the host to send or receive traffic. For port name, a default will be assigned if left blank.

Container port	Protocol	Port name	App protocol
3000	TCP	port-fargate-3000	HTTP

Remove

Add port mapping

Create it final output.



nodejs-server-demo-td:1

Overview Info

ARN arn:aws:ecs:ap-south-1:199316303204:task-definition/nodejs-server-demo-td:1	**Status** ACTIVE	**Time created** January 21, 2025 at 02:21 (UTC+5:30)	**App environment** Fargate
Task role -	**Task execution role** ecsTaskExecutionRole	**Operating system/Architecture** Linux/X86_64	**Network mode** awsvpc
Fault injection Turned off			

Task size

Task CPU 512 units (0.5 vCPU)

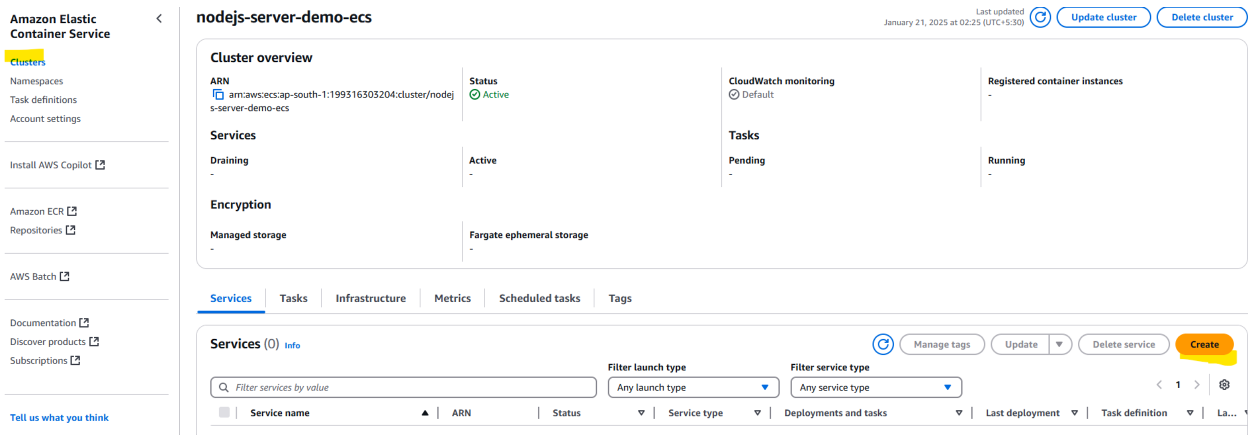
Task CPU maximum allocation for containers

Task memory 1,024 MiB (1 GB)

Task memory maximum allocation for container memory reservation

Create ECS Service:

Go to the cluster and down the service create.



nodejs-server-demo-ecs

Last updated January 21, 2025 at 02:25 (UTC+5:30)

Cluster overview

ARN arn:aws:ecs:ap-south-1:199316303204:cluster/nodejs-server-demo-ecs	**Status** Active	**CloudWatch monitoring** Default	**Registered container instances** -

Services

Draining -	**Active** -

Tasks

Pending -	**Running** -

Encryption

Managed storage -	**Fargate ephemeral storage** -

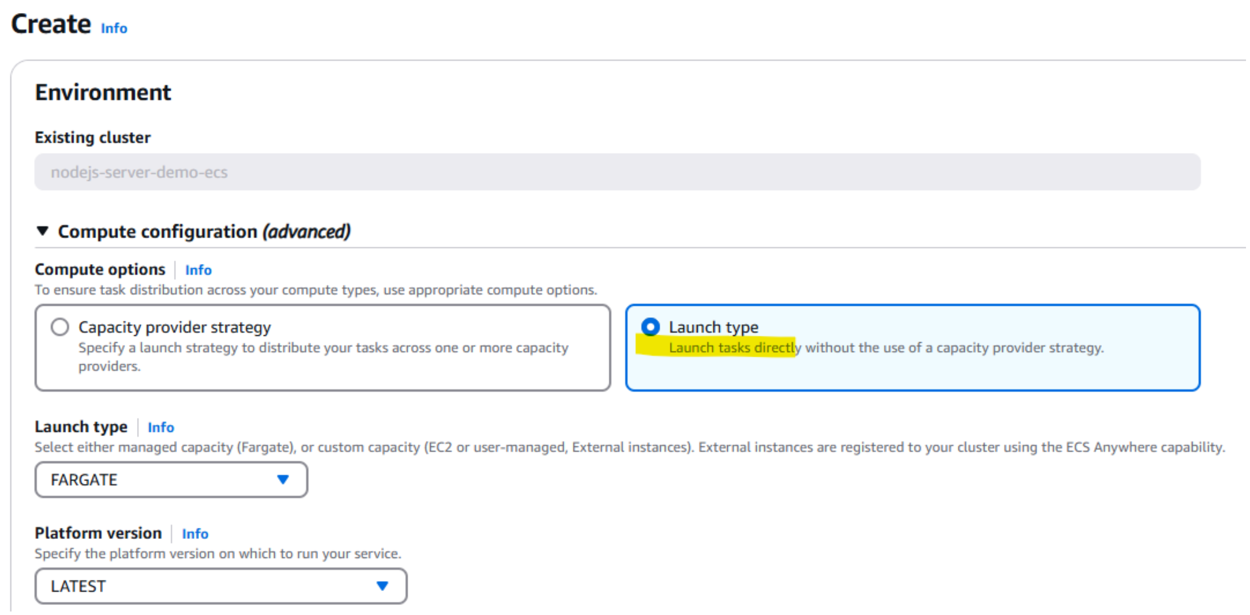
Services (0) Info

Filter services by value

Filter launch type: Any launch type

Filter service type: Any service type

Service name	ARN	Status	Service type	Deployments and tasks	Last deployment	Task definition	La…



Create Info

Environment

Existing cluster

nodejs-server-demo-ecs

▼ **Compute configuration *(advanced)***

Compute options Info

To ensure task distribution across your compute types, use appropriate compute options.

- ○ Capacity provider strategy
  Specify a launch strategy to distribute your tasks across one or more capacity providers.
- ● Launch type
  Launch tasks directly without the use of a capacity provider strategy.

Launch type Info

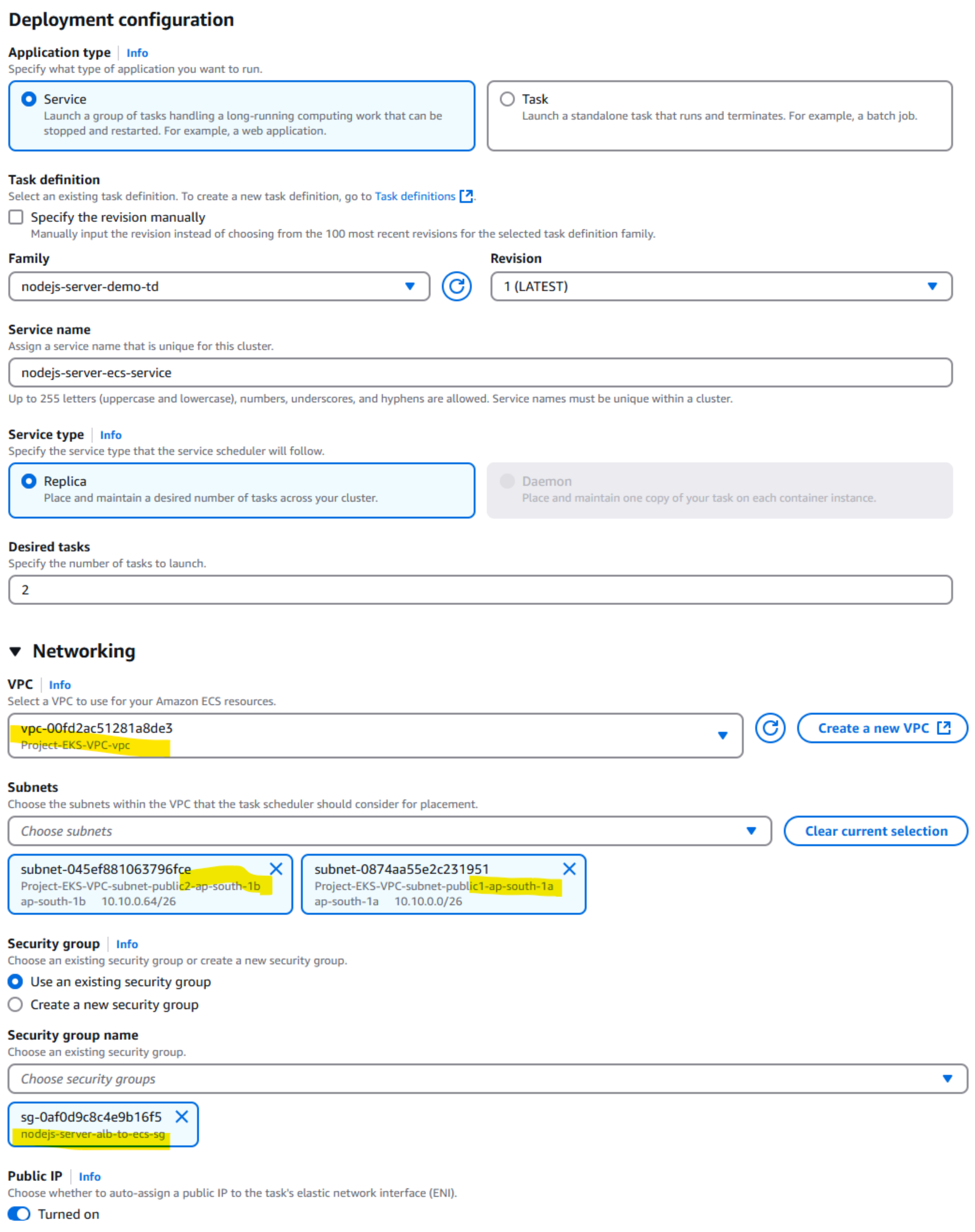
Select either managed capacity (Fargate), or custom capacity (EC2 or user-managed, External instances). External instances are registered to your cluster using the ECS Anywhere capability.

FARGATE

Platform version Info

Specify the platform version on which to run your service.

LATEST



Deployment configuration

Application type Info
Specify what type of application you want to run.

[✓ Service] Launch a group of tasks handling a long-running computing work that can be stopped and restarted. For example, a web application.

Task Launch a standalone task that runs and terminates. For example, a batch job.

Task definition
Select an existing task definition. To create a new task definition, go to Task definitions.

☐ Specify the revision manually
Manually input the revision instead of choosing from the 100 most recent revisions for the selected task definition family.

Family	nodejs-server-demo-td
Revision	1 (LATEST)

Service name
Assign a service name that is unique for this cluster.

nodejs-server-ecs-service

Up to 255 letters (uppercase and lowercase), numbers, underscores, and hyphens are allowed. Service names must be unique within a cluster.

Service type Info
Specify the service type that the service scheduler will follow.

[✓ Replica] Place and maintain a desired number of tasks across your cluster.

Daemon Place and maintain one copy of your task on each container instance.

Desired tasks
Specify the number of tasks to launch.

2

Networking

VPC Info
Select a VPC to use for your Amazon ECS resources.

vpc-00fd2ac51281a8de3
Project-EKS-VPC-vpc

Create a new VPC

Subnets
Choose the subnets within the VPC that the task scheduler should consider for placement.

Choose subnets

Clear current selection

subnet-045ef881063796fce
Project-EKS-VPC-subnet-public2-ap-south-1b
ap-south-1b 10.10.0.64/26

subnet-0874aa55e2c231951
Project-EKS-VPC-subnet-public1-ap-south-1a
ap-south-1a 10.10.0.0/26

Security group Info
Choose an existing security group or create a new security group.

[✓ Use an existing security group]
Create a new security group

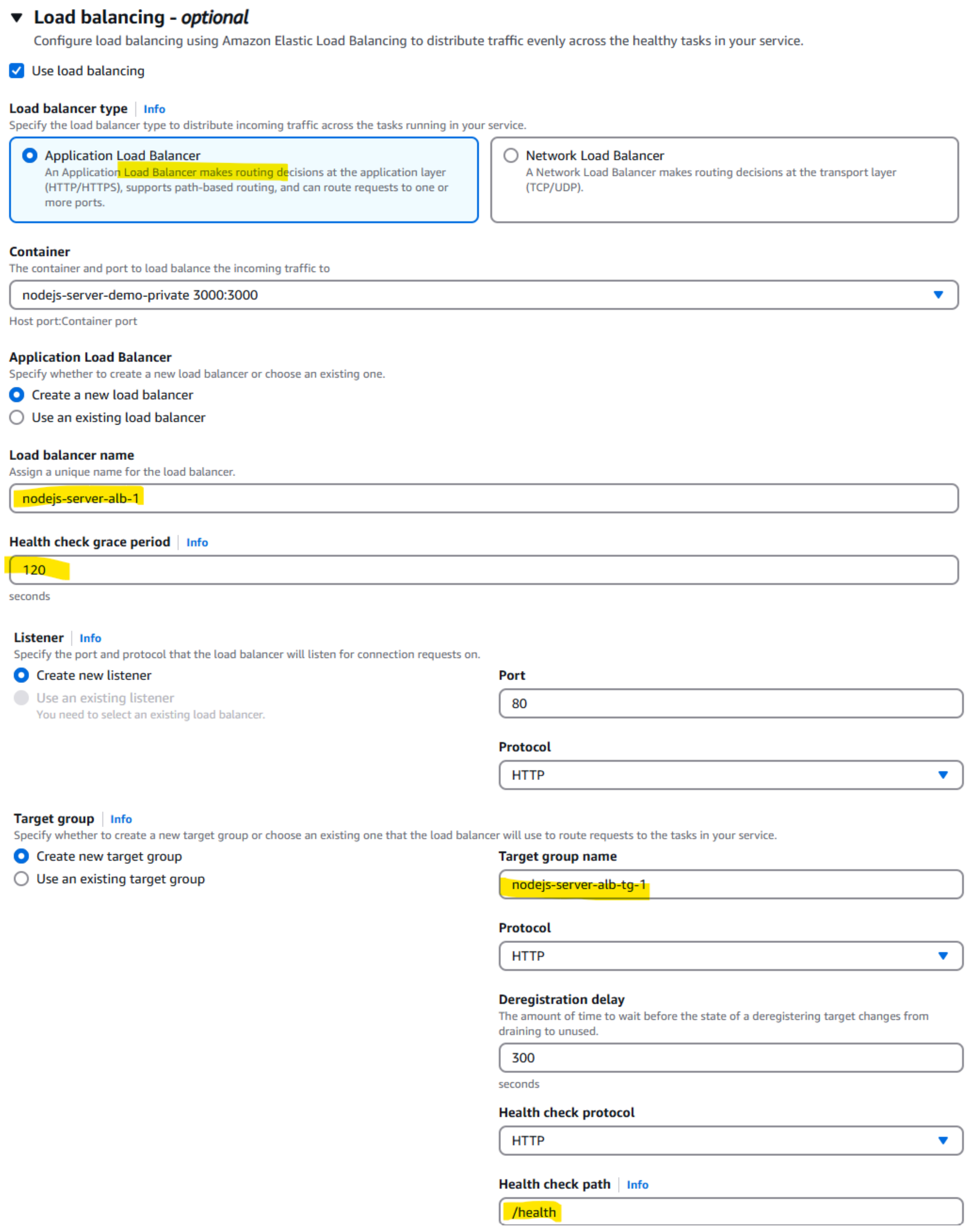
Security group name
Choose an existing security group.

Choose security groups

sg-0af0d9c8c4e9b16f5
nodejs-server-alb-to-ecs-sg

Public IP Info
Choose whether to auto-assign a public IP to the task's elastic network interface (ENI).

Turned on



▼ **Load balancing** - *optional*

Configure load balancing using Amazon Elastic Load Balancing to distribute traffic evenly across the healthy tasks in your service.

✓ Use load balancing

Load balancer type | Info

Specify the load balancer type to distribute incoming traffic across the tasks running in your service.

● Application Load Balancer
An Application Load Balancer makes routing decisions at the application layer (HTTP/HTTPS), supports path-based routing, and can route requests to one or more ports.

○ Network Load Balancer
A Network Load Balancer makes routing decisions at the transport layer (TCP/UDP).

Container

The container and port to load balance the incoming traffic to

nodejs-server-demo-private 3000:3000

Host port:Container port

Application Load Balancer

Specify whether to create a new load balancer or choose an existing one.

● Create a new load balancer
○ Use an existing load balancer

Load balancer name

Assign a unique name for the load balancer.

nodejs-server-alb-1

Health check grace period | Info

120

seconds

Listener | Info

Specify the port and protocol that the load balancer will listen for connection requests on.

● Create new listener
○ Use an existing listener
You need to select an existing load balancer.

Port

80

Protocol

HTTP

Target group | Info

Specify whether to create a new target group or choose an existing one that the load balancer will use to route requests to the tasks in your service.

● Create new target group
○ Use an existing target group

Target group name

nodejs-server-alb-tg-1

Protocol

HTTP

Deregistration delay

The amount of time to wait before the state of a deregistering target changes from draining to unused.

300

seconds

Health check protocol

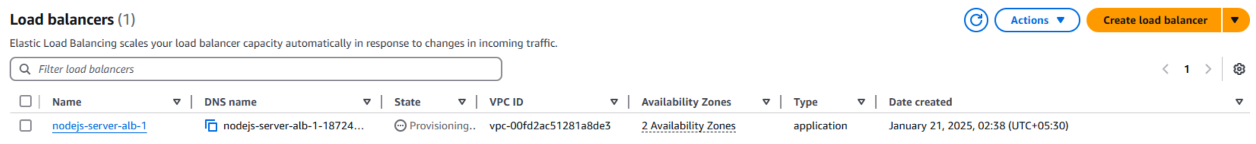
HTTP

Health check path | Info

/health

Create It.

Just Check the Load balancer created or not created it’s fine.

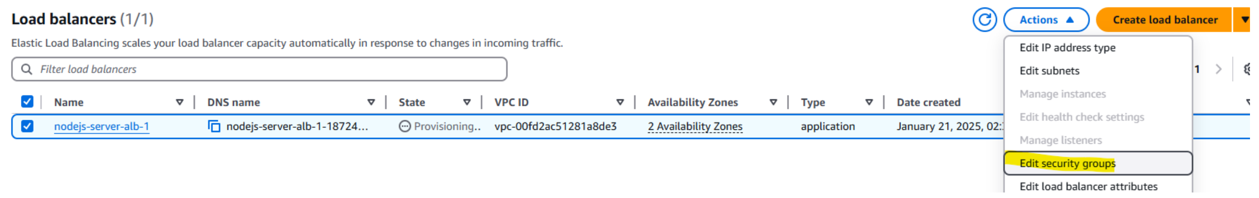


Load balancers (1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
nodejs-server-alb-1	nodejs-server-alb-1-18724...	Provisioning..	vpc-00fd2ac51281a8de3	2 Availability Zones	application	January 21, 2025, 02:38 (UTC+05:30)



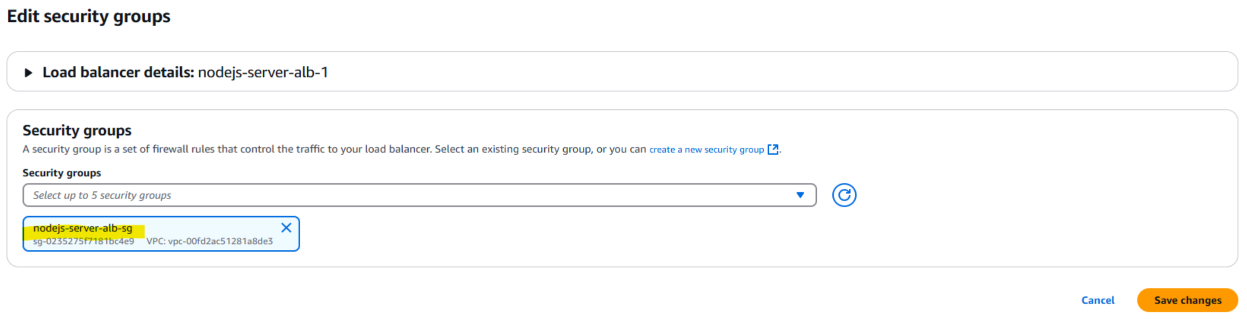
Load balancers (1/1)

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
✓ nodejs-server-alb-1	nodejs-server-alb-1-18724...	Provisioning..	vpc-00fd2ac51281a8de3	2 Availability Zones	application	January 21, 2025, 02:

Change the Security Group.



Edit security groups

Load balancer details: nodejs-server-alb-1

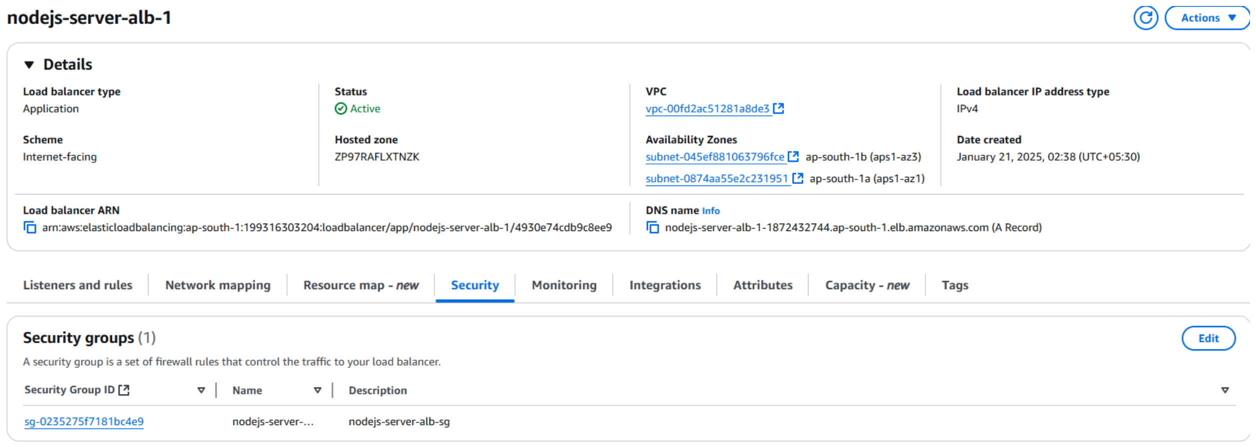
Security groups

A security group is a set of firewall rules that control the traffic to your load balancer. Select an existing security group, or you can create a new security group

Security groups

Select up to 5 security groups

nodejs-server-alb-sg
sg-0235275f7181bc4e9 VPC: vpc-00fd2ac51281a8de3



nodejs-server-alb-1

Details

Load balancer type Application	**Status** Active	**VPC** vpc-00fd2ac51281a8de3	**Load balancer IP address type** IPv4
Scheme Internet-facing	**Hosted zone** ZP97RAFLXTNZK	**Availability Zones** subnet-045ef881063796fce ap-south-1b (aps1-az3); subnet-0874aa55e2c231951 ap-south-1a (aps1-az1)	**Date created** January 21, 2025, 02:38 (UTC+05:30)

Load balancer ARN
arn:aws:elasticloadbalancing:ap-south-1:199316303204:loadbalancer/app/nodejs-server-alb-1/4930e74cdb9c8ee9

DNS name Info
nodejs-server-alb-1-1872432744.ap-south-1.elb.amazonaws.com (A Record)

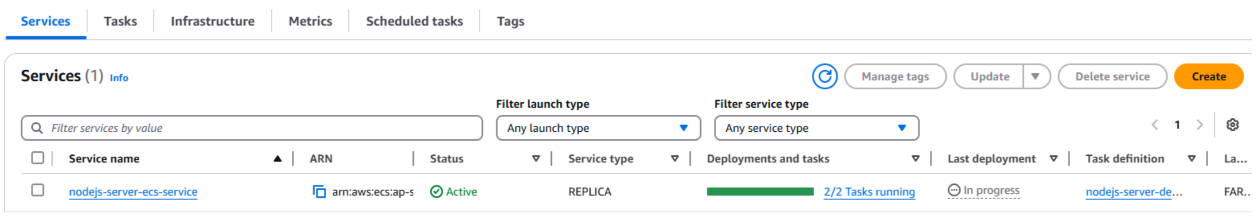
Listeners and rules | Network mapping | Resource map - *new* | Security | Monitoring | Integrations | Attributes | Capacity - *new* | Tags

Security groups (1)

A security group is a set of firewall rules that control the traffic to your load balancer.

Security Group ID	Name	Description
sg-0235275f7181bc4e9	nodejs-server-...	nodejs-server-alb-sg

Once Go to check the ECS service Service is created or not.



Services | Tasks | Infrastructure | Metrics | Scheduled tasks | Tags

Services (1) Info

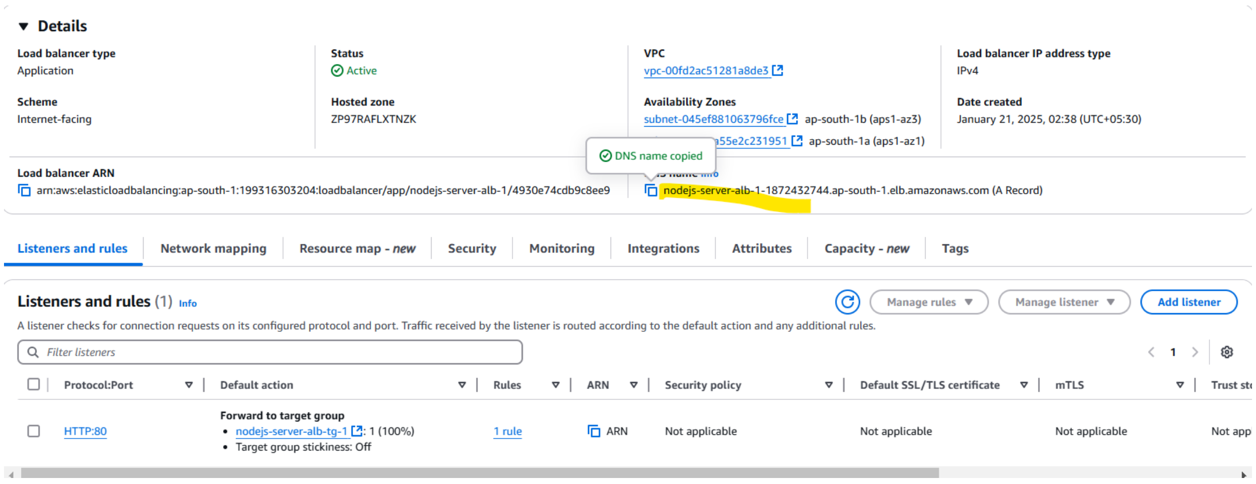
Filter services by value

Filter launch type: Any launch type

Filter service type: Any service type

Service name	ARN	Status	Service type	Deployments and tasks	Last deployment	Task definition	La...
nodejs-server-ecs-service	arn:aws:ecs:ap-s	Active	REPLICA	2/2 Tasks running	In progress	nodejs-server-de...	FAR..

After that the Load Balancer copy the DNS and paste the webpage.



▼ Details

Load balancer type	Status	VPC	Load balancer IP address type
Application	Active	vpc-00fd2ac51281a8de3	IPv4
Scheme	**Hosted zone**	**Availability Zones**	**Date created**
Internet-facing	ZP97RAFLXTNZK	subnet-045ef881063796fce ap-south-1b (aps1-az3)	January 21, 2025, 02:38 (UTC+05:30)
		55e2c231951 ap-south-1a (aps1-az1)	

DNS name copied

Load balancer ARN
arn:aws:elasticloadbalancing:ap-south-1:199316303204:loadbalancer/app/nodejs-server-alb-1/4930e74cdb9c8ee9

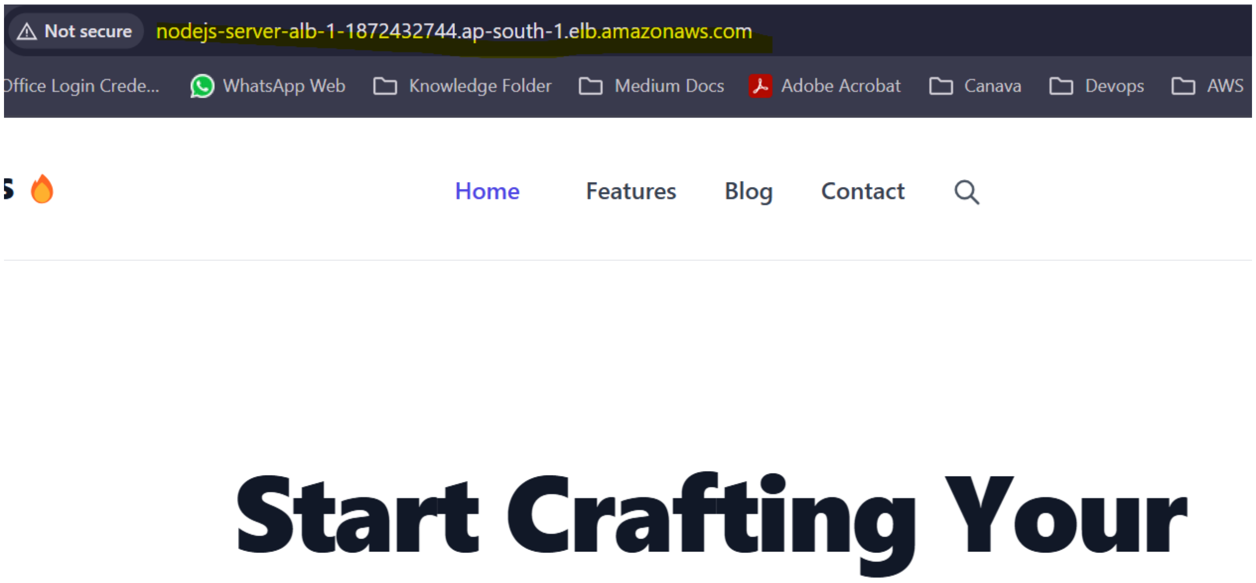
nodejs-server-alb-1-1872432744.ap-south-1.elb.amazonaws.com (A Record)

Listeners and rules | Network mapping | Resource map - *new* | Security | Monitoring | Integrations | Attributes | Capacity - *new* | Tags

Listeners and rules (1) Info

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust st
HTTP:80	Forward to target group • nodejs-server-alb-tg-1: 1 (100%) • Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not app



Not secure nodejs-server-alb-1-1872432744.ap-south-1.elb.amazonaws.com

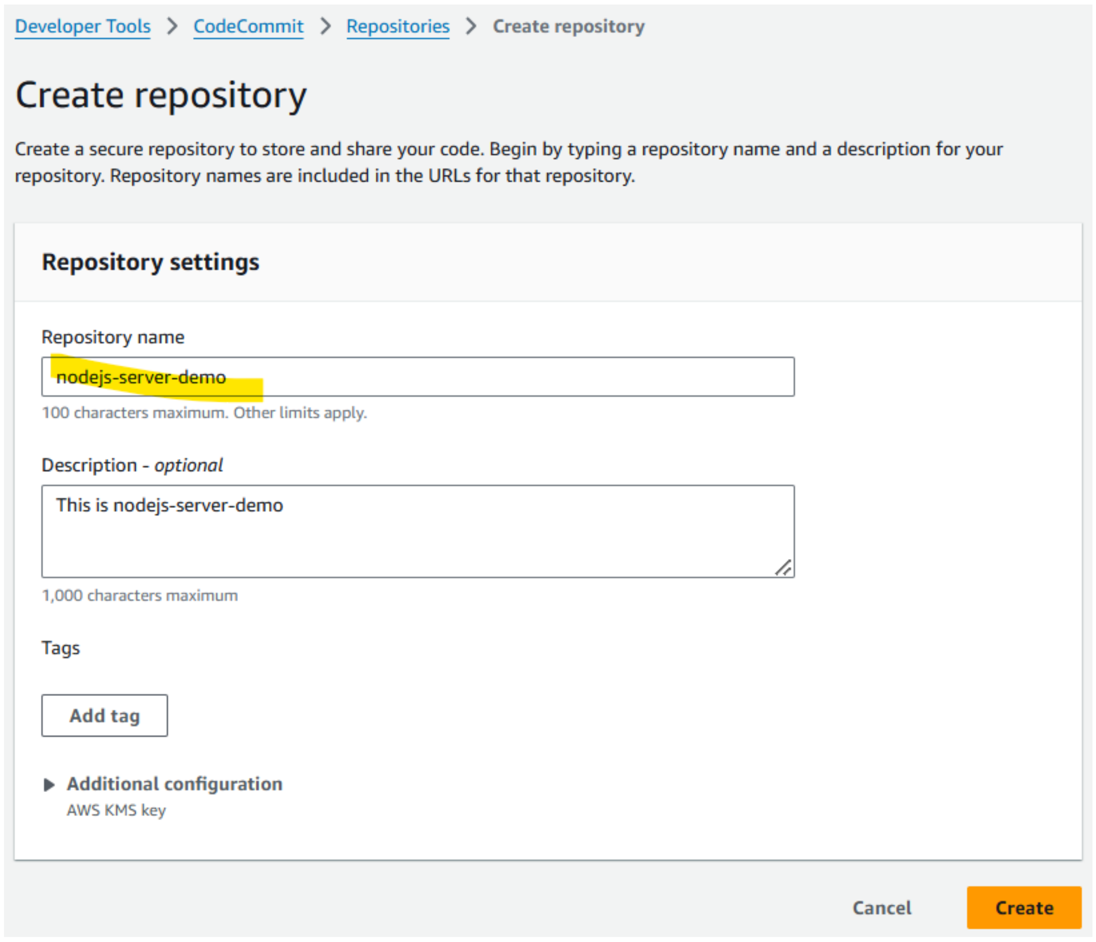
It’s Working.

And let go back to ECS service.

Check all are ok or not.

Create Code Commit Repo:

Open the Code commit service and create new repository.



Developer Tools > CodeCommit > Repositories > Create repository

Create repository

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

Repository settings

Repository name: nodejs-server-demo

100 characters maximum. Other limits apply.

Description - *optional*: This is nodejs-server-demo

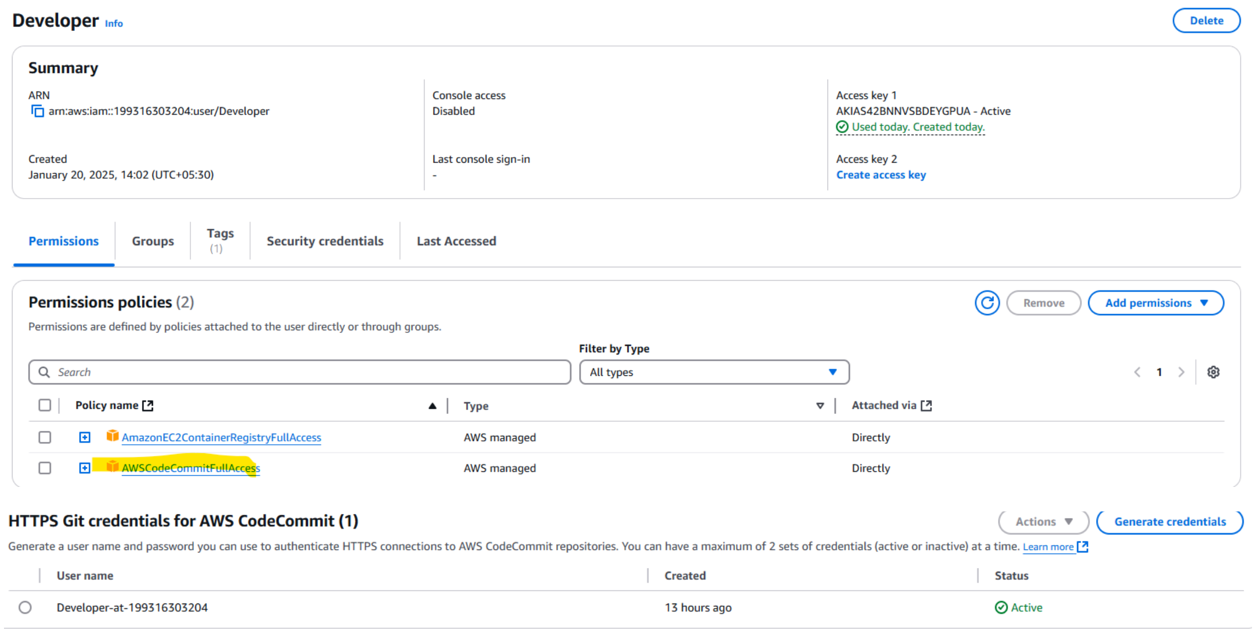
1,000 characters maximum

Tags

Push Code to Code Commit Repo:

First Open the IAM service give the access for code commit.

And create the HTTPS Git credentials.



Developer Info

Delete

Summary

ARN arn:aws:iam::199316303204:user/Developer	Console access Disabled	Access key 1 AKIAS42BNNVSBDEYGPUA - Active Used today. Created today.
Created January 20, 2025, 14:02 (UTC+05:30)	Last console sign-in -	Access key 2 Create access key

Permissions | Groups | Tags (1) | Security credentials | Last Accessed

Permissions policies (2)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type: All types

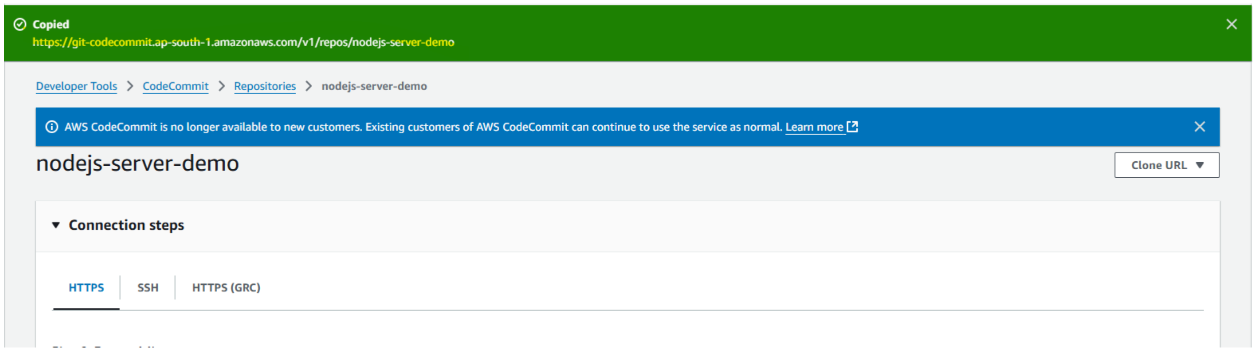
Policy name	Type	Attached via
AmazonEC2ContainerRegistryFullAccess	AWS managed	Directly
AWSCodeCommitFullAccess	AWS managed	Directly

HTTPS Git credentials for AWS CodeCommit (1)

Generate a user name and password you can use to authenticate HTTPS connections to AWS CodeCommit repositories. You can have a maximum of 2 sets of credentials (active or inactive) at a time. Learn more

User name	Created	Status
Developer-at-199316303204	13 hours ago	Active

After return into Code commit console get the clone https url.



Copied
https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/nodejs-server-demo

Developer Tools > CodeCommit > Repositories > nodejs-server-demo

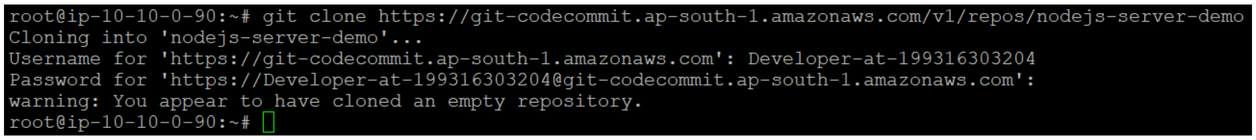
AWS CodeCommit is no longer available to new customers. Existing customers of AWS CodeCommit can continue to use the service as normal. Learn more

nodejs-server-demo

Clone URL
Connection steps
HTTPS
SSH
HTTPS (GRC)

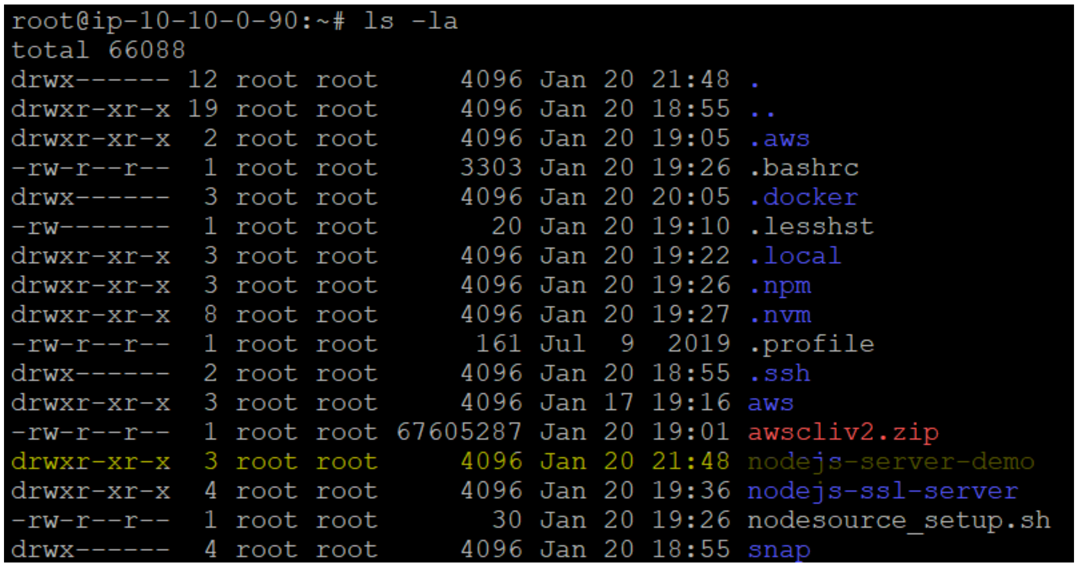
After go the ssh clone the https url in git.

git clone https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/nodejs-server-demo



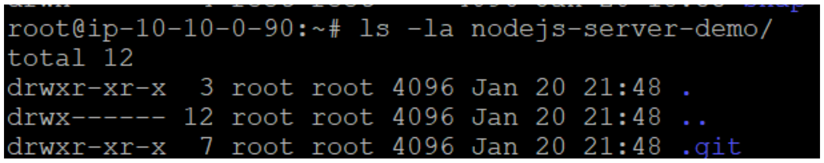
```
root@ip-10-10-0-90:~# git clone https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/nodejs-server-demo
Cloning into 'nodejs-server-demo'...
Username for 'https://git-codecommit.ap-south-1.amazonaws.com': Developer-at-199316303204
Password for 'https://Developer-at-199316303204@git-codecommit.ap-south-1.amazonaws.com':
warning: You appear to have cloned an empty repository.
root@ip-10-10-0-90:~#
```

ls -la



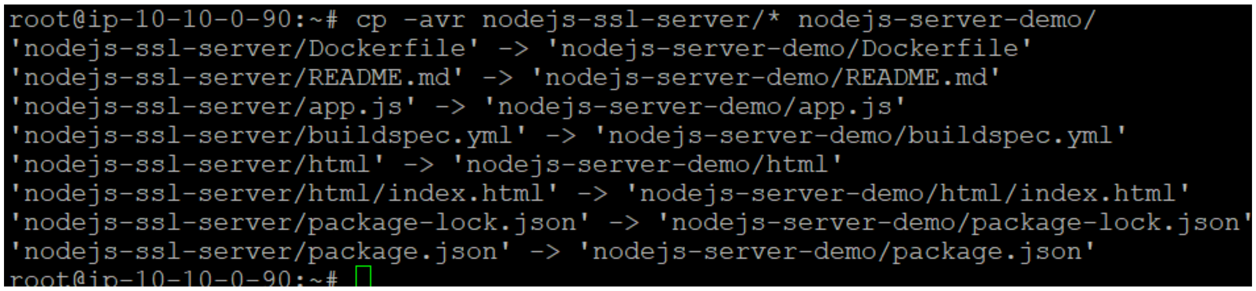
```
root@ip-10-10-0-90:~# ls -la
total 66088
drwx------ 12 root root     4096 Jan 20 21:48 .
drwxr-xr-x 19 root root     4096 Jan 20 18:55 ..
drwxr-xr-x  2 root root     4096 Jan 20 19:05 .aws
-rw-r--r--  1 root root     3303 Jan 20 19:26 .bashrc
drwx------  3 root root     4096 Jan 20 20:05 .docker
-rw-------  1 root root       20 Jan 20 19:10 .lesshst
drwxr-xr-x  3 root root     4096 Jan 20 19:22 .local
drwxr-xr-x  3 root root     4096 Jan 20 19:26 .npm
drwxr-xr-x  8 root root     4096 Jan 20 19:27 .nvm
-rw-r--r--  1 root root      161 Jul  9  2019 .profile
drwx------  2 root root     4096 Jan 20 18:55 .ssh
drwxr-xr-x  3 root root     4096 Jan 17 19:16 aws
-rw-r--r--  1 root root 67605287 Jan 20 19:01 awscliv2.zip
drwxr-xr-x  3 root root     4096 Jan 20 21:48 nodejs-server-demo
drwxr-xr-x  4 root root     4096 Jan 20 19:36 nodejs-ssl-server
-rw-r--r--  1 root root       30 Jan 20 19:26 nodesource_setup.sh
drwx------  4 root root     4096 Jan 20 18:55 snap
```

ls -la nodejs-server-demo/



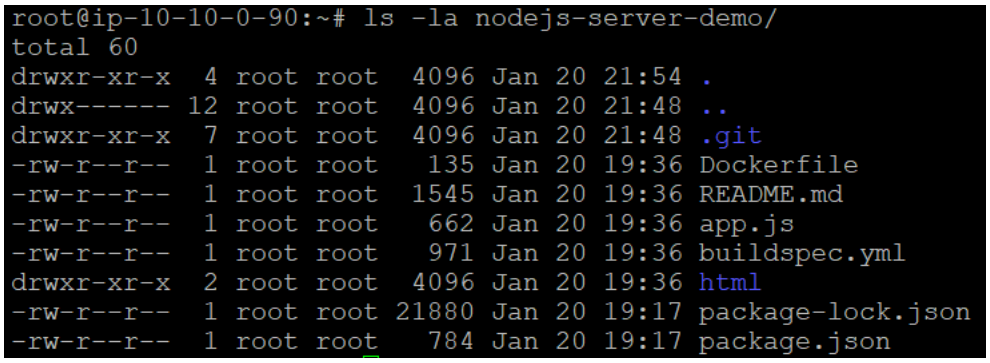
```
root@ip-10-10-0-90:~# ls -la nodejs-server-demo/
total 12
drwxr-xr-x  3 root root 4096 Jan 20 21:48 .
drwx------ 12 root root 4096 Jan 20 21:48 ..
drwxr-xr-x  7 root root 4096 Jan 20 21:48 .git
```

cp -avr nodejs-ssl-server/* nodejs-server-demo/



```
root@ip-10-10-0-90:~# cp -avr nodejs-ssl-server/* nodejs-server-demo/
'nodejs-ssl-server/Dockerfile' -> 'nodejs-server-demo/Dockerfile'
'nodejs-ssl-server/README.md' -> 'nodejs-server-demo/README.md'
'nodejs-ssl-server/app.js' -> 'nodejs-server-demo/app.js'
'nodejs-ssl-server/buildspec.yml' -> 'nodejs-server-demo/buildspec.yml'
'nodejs-ssl-server/html' -> 'nodejs-server-demo/html'
'nodejs-ssl-server/html/index.html' -> 'nodejs-server-demo/html/index.html'
'nodejs-ssl-server/package-lock.json' -> 'nodejs-server-demo/package-lock.json'
'nodejs-ssl-server/package.json' -> 'nodejs-server-demo/package.json'
root@ip-10-10-0-90:~#
```

ls -la nodejs-server-demo/



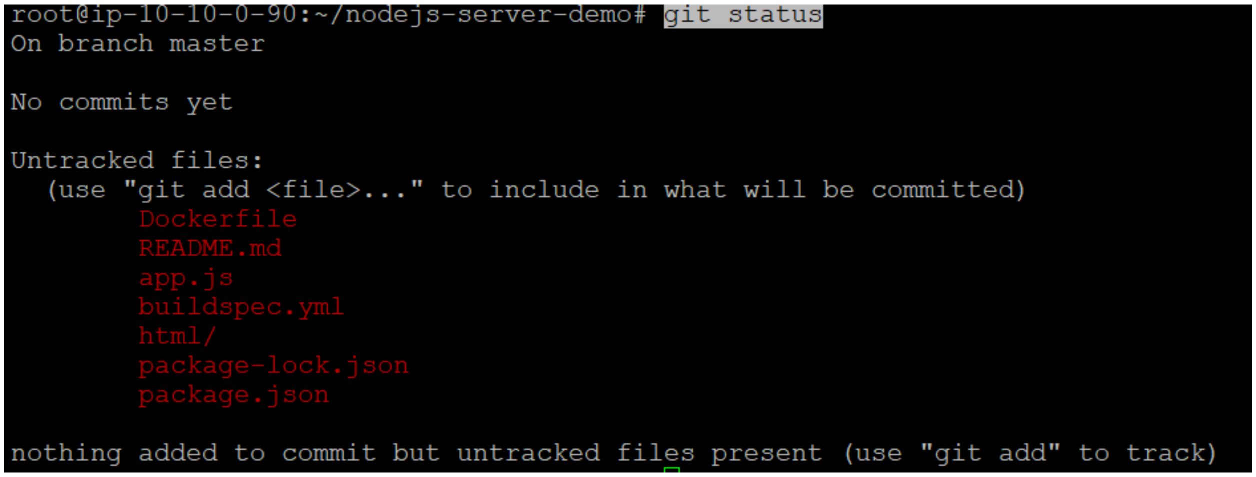
```
root@ip-10-10-0-90:~# ls -la nodejs-server-demo/
total 60
drwxr-xr-x  4 root root  4096 Jan 20 21:54 .
drwx------ 12 root root  4096 Jan 20 21:48 ..
drwxr-xr-x  7 root root  4096 Jan 20 21:48 .git
-rw-r--r--  1 root root   135 Jan 20 19:36 Dockerfile
-rw-r--r--  1 root root  1545 Jan 20 19:36 README.md
-rw-r--r--  1 root root   662 Jan 20 19:36 app.js
-rw-r--r--  1 root root   971 Jan 20 19:36 buildspec.yml
drwxr-xr-x  2 root root  4096 Jan 20 19:36 html
-rw-r--r--  1 root root 21880 Jan 20 19:17 package-lock.json
-rw-r--r--  1 root root   784 Jan 20 19:17 package.json
```

cd nodejs-server-demo/

nano buildspec.yml

```
version: 0.2


phases:

  pre_build:

    commands:

      - echo Logging in to Amazon ECR...

      - aws --version

      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --username AWS --password-
stdin $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.a>

      -
REPOSITORY_URI=$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME

  build:

    commands:

      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $REPOSITORY_URI:$IMAGE_TAG .
      - docker tag $REPOSITORY_URI:$IMAGE_TAG $REPOSITORY_URI:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker images...
      - docker push $REPOSITORY_URI:$IMAGE_TAG
      - echo Writing image definitions file...
      - printf '[{"name":"%s","imageUri":"%s"}]' $CONTAINER_NAME $REPOSITORY_URI:$IMAGE_TAG >
imagedefinitions.json


artifacts:
    files: imagedefinitions.json
```

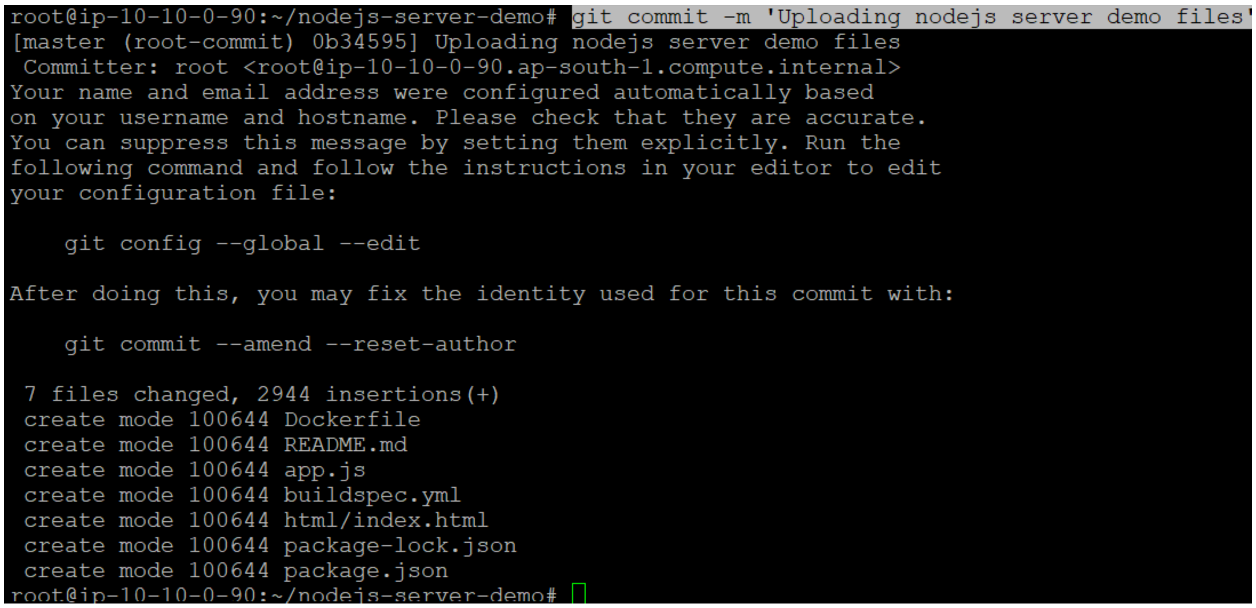
git status



```
root@ip-10-10-0-90:~/nodejs-server-demo# git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Dockerfile
        README.md
        app.js
        buildspec.yml
        html/
        package-lock.json
        package.json

nothing added to commit but untracked files present (use "git add" to track)
```

git add -- .

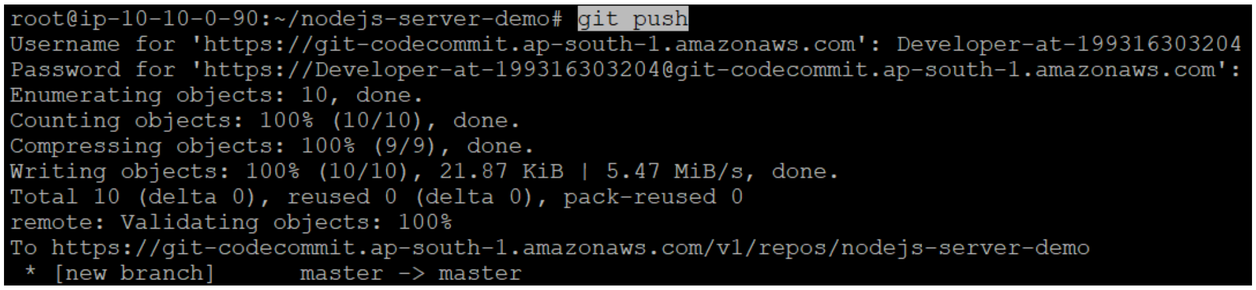
git commit -m 'Uploading nodejs server demo files'



```
root@ip-10-10-0-90:~/nodejs-server-demo# git commit -m 'Uploading nodejs server demo files'
[master (root-commit) 0b34595] Uploading nodejs server demo files
 Committer: root <root@ip-10-10-0-90.ap-south-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 7 files changed, 2944 insertions(+)
 create mode 100644 Dockerfile
 create mode 100644 README.md
 create mode 100644 app.js
 create mode 100644 buildspec.yml
 create mode 100644 html/index.html
 create mode 100644 package-lock.json
 create mode 100644 package.json
root@ip-10-10-0-90:~/nodejs-server-demo#
```

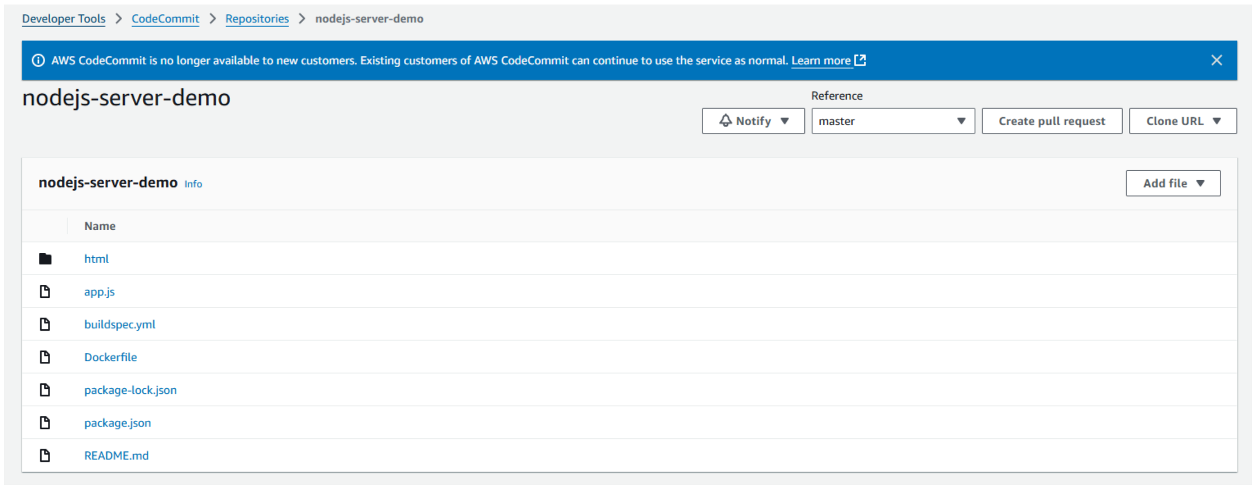
git push

credential use for HTTPS git credential in IAM user.



```
root@ip-10-10-0-90:~/nodejs-server-demo# git push
Username for 'https://git-codecommit.ap-south-1.amazonaws.com': Developer-at-199316303204
Password for 'https://Developer-at-199316303204@git-codecommit.ap-south-1.amazonaws.com':
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Compressing objects: 100% (9/9), done.
Writing objects: 100% (10/10), 21.87 KiB | 5.47 MiB/s, done.
Total 10 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
To https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/nodejs-server-demo
 * [new branch]      master -> master
```

Check the Code commit repository updated or not.



Developer Tools > CodeCommit > Repositories > nodejs-server-demo

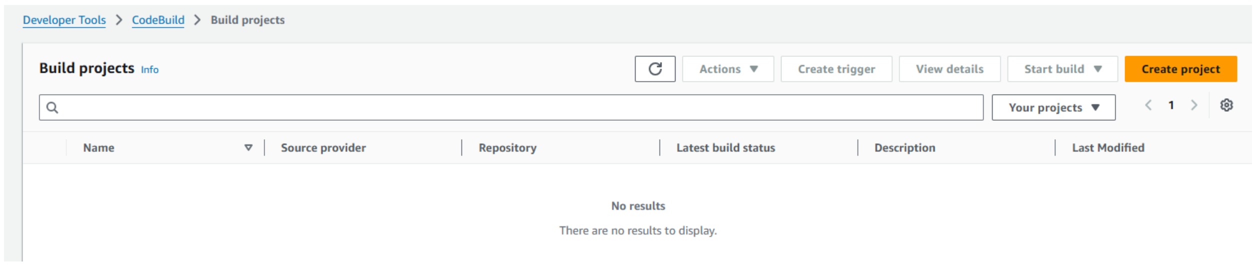
AWS CodeCommit is no longer available to new customers. Existing customers of AWS CodeCommit can continue to use the service as normal. Learn more

nodejs-server-demo

nodejs-server-demo

Name
html
app.js
buildspec.yml
Dockerfile
package-lock.json
package.json
README.md

Create Code Build Project:


Developer Tools > CodeBuild > Build projects

Build projects Info

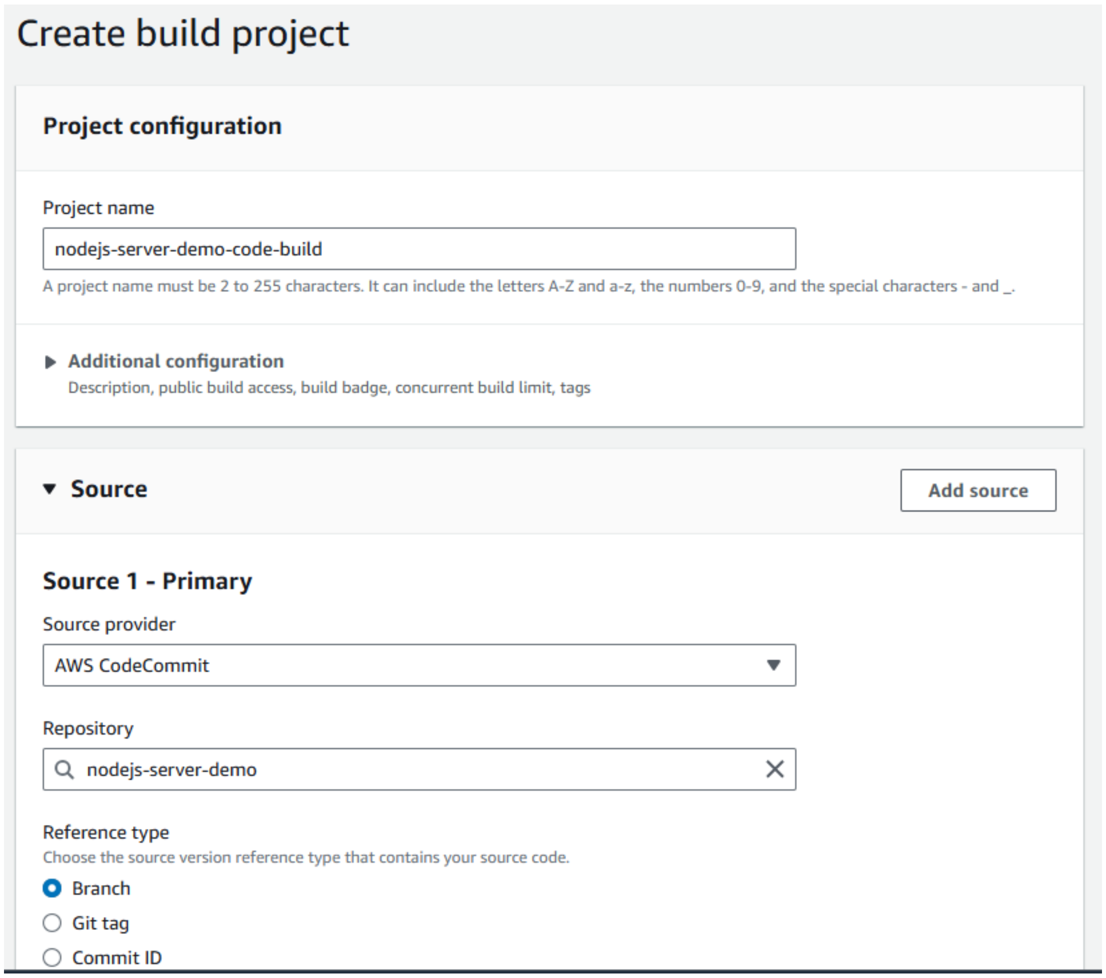
Actions | Create trigger | View details | Start build | Create project

Your projects

Name	Source provider	Repository	Latest build status	Description	Last Modified

No results

There are no results to display.


Create build project

Project configuration

Project name

nodejs-server-demo-code-build

A project name must be 2 to 255 characters. It can include the letters A-Z and a-z, the numbers 0-9, and the special characters - and _.

Additional configuration

Description, public build access, build badge, concurrent build limit, tags

Source

Add source

Source 1 - Primary

Source provider

AWS CodeCommit

Repository

nodejs-server-demo

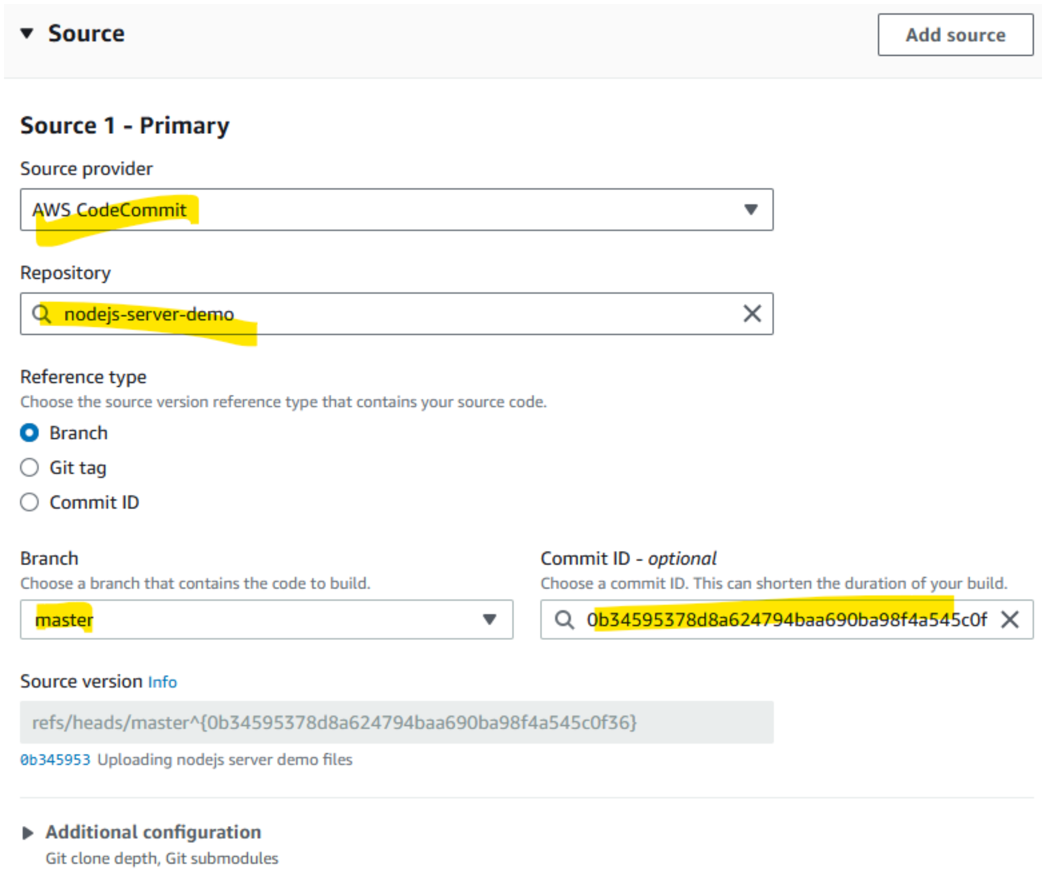
Reference type

Choose the source version reference type that contains your source code.

● Branch

○ Git tag

○ Commit ID



▼ Source

Add source

Source 1 - Primary

Source provider

AWS CodeCommit

Repository

nodejs-server-demo

Reference type

Choose the source version reference type that contains your source code.

- [✓ Branch]
- Git tag
- Commit ID

Branch

Choose a branch that contains the code to build.

master

Commit ID - *optional*

Choose a commit ID. This can shorten the duration of your build.

0b34595378d8a624794baa690ba98f4a545c0f

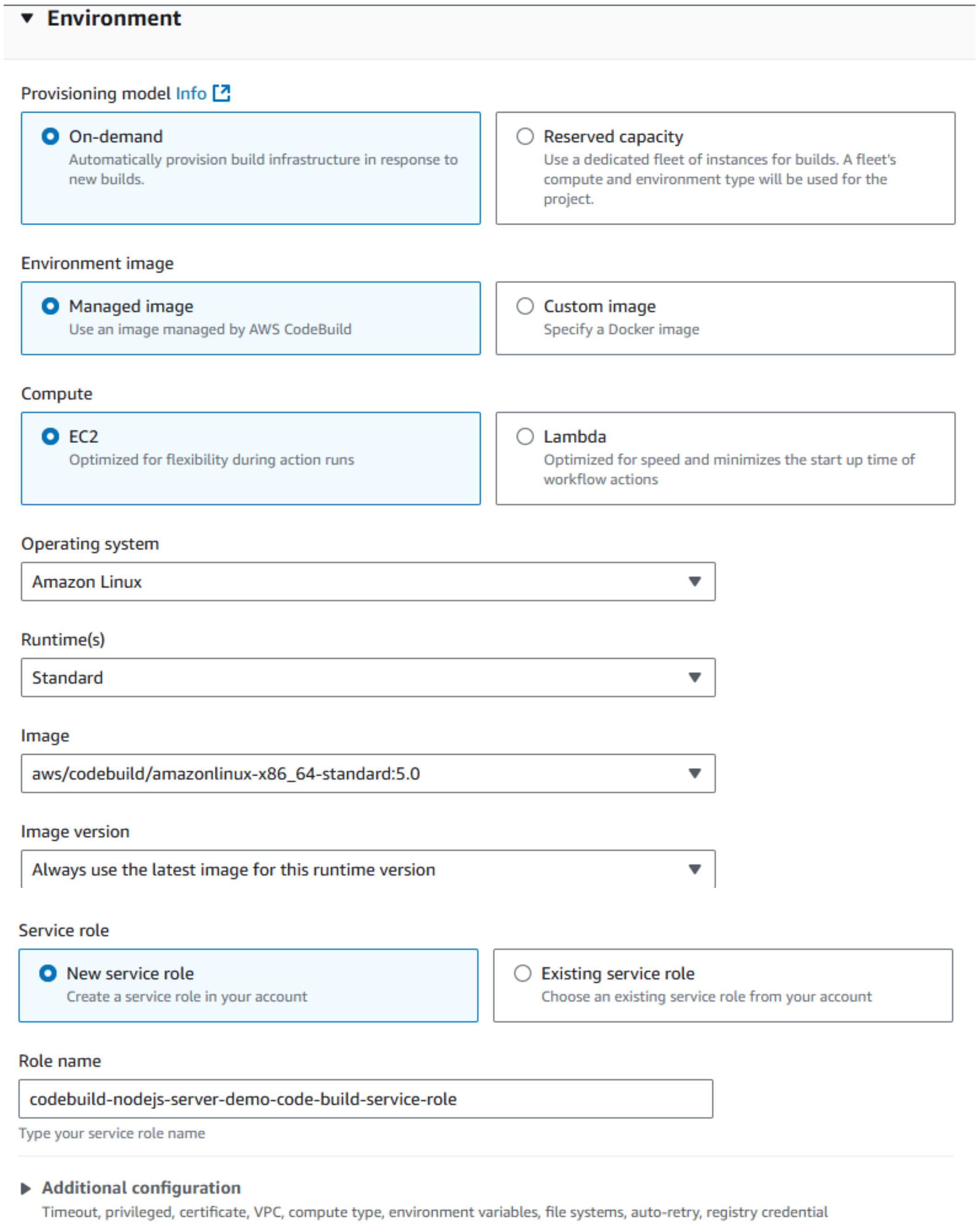
Source version Info

refs/heads/master^{0b34595378d8a624794baa690ba98f4a545c0f36}

0b345953 Uploading nodejs server demo files

▶ Additional configuration

Git clone depth, Git submodules



▼ **Environment**

Provisioning model Info

[✓ On-demand] Automatically provision build infrastructure in response to new builds.

Reserved capacity: Use a dedicated fleet of instances for builds. A fleet's compute and environment type will be used for the project.

Environment image

[✓ Managed image] Use an image managed by AWS CodeBuild

Custom image: Specify a Docker image

Compute

[✓ EC2] Optimized for flexibility during action runs

Lambda: Optimized for speed and minimizes the start up time of workflow actions

Operating system: Amazon Linux

Runtime(s): Standard

Image: aws/codebuild/amazonlinux-x86_64-standard:5.0

Image version: Always use the latest image for this runtime version

Service role

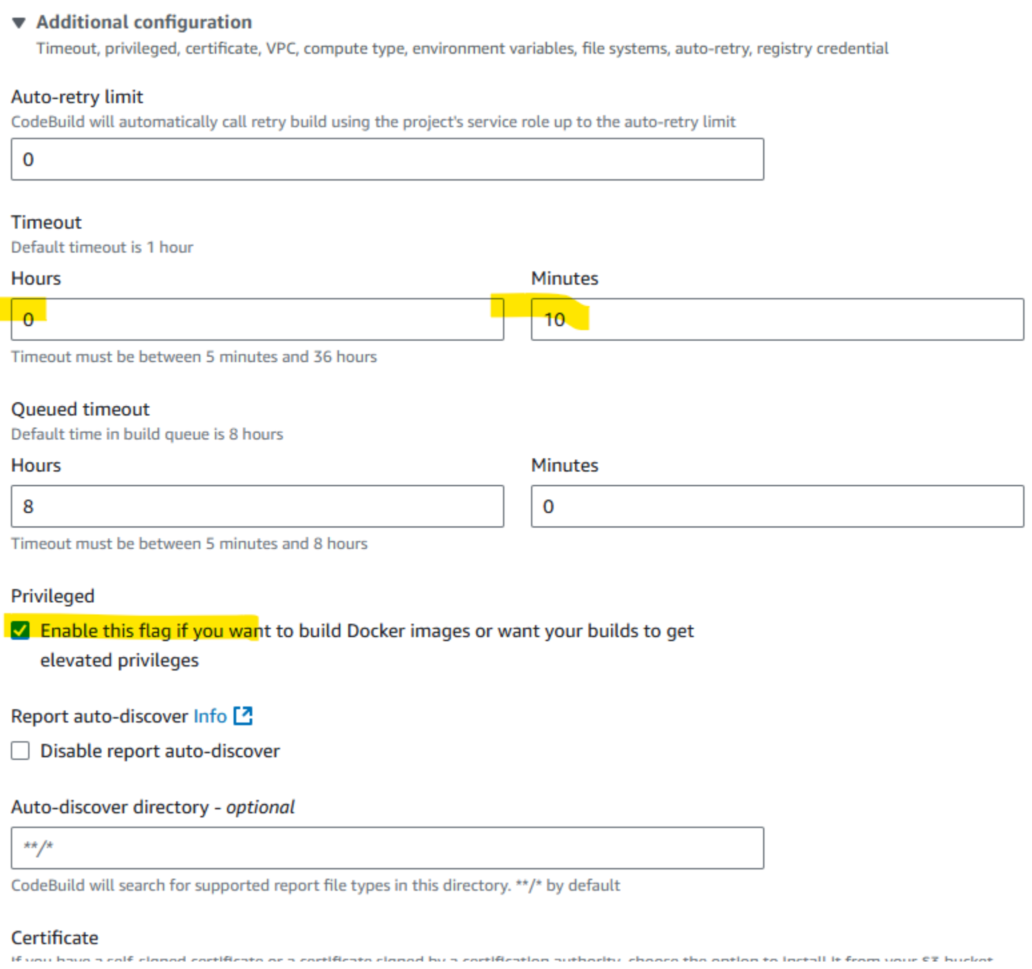
[✓ New service role] Create a service role in your account

Existing service role: Choose an existing service role from your account

Role name: codebuild-nodejs-server-demo-code-build-service-role

Type your service role name

▶ **Additional configuration**
Timeout, privileged, certificate, VPC, compute type, environment variables, file systems, auto-retry, registry credential


▼ Additional configuration
Timeout, privileged, certificate, VPC, compute type, environment variables, file systems, auto-retry, registry credential

Auto-retry limit
CodeBuild will automatically call retry build using the project's service role up to the auto-retry limit
0

Timeout
Default timeout is 1 hour

Hours	Minutes
0	10

Timeout must be between 5 minutes and 36 hours

Queued timeout
Default time in build queue is 8 hours

Hours	Minutes
8	0

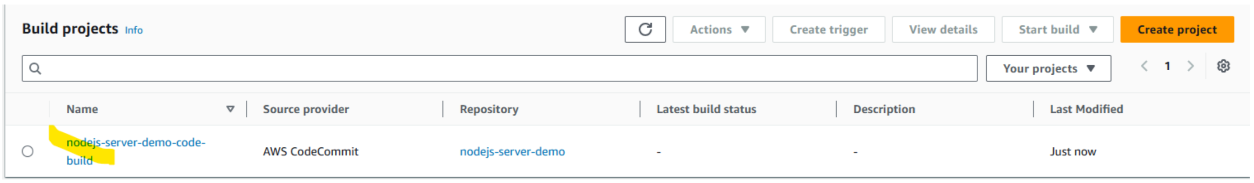
Timeout must be between 5 minutes and 8 hours

Privileged
✓ Enable this flag if you want to build Docker images or want your builds to get elevated privileges

Report auto-discover Info
☐ Disable report auto-discover

Auto-discover directory - *optional*
**/*
CodeBuild will search for supported report file types in this directory. **/* by default

Certificate

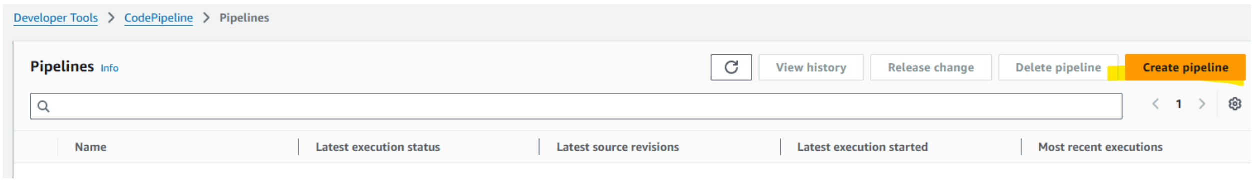

Build projects Info

	Name	Source provider	Repository	Latest build status	Description	Last Modified
○	nodejs-server-demo-code-build	AWS CodeCommit	nodejs-server-demo	-	-	Just now

Finally Create the Build project.

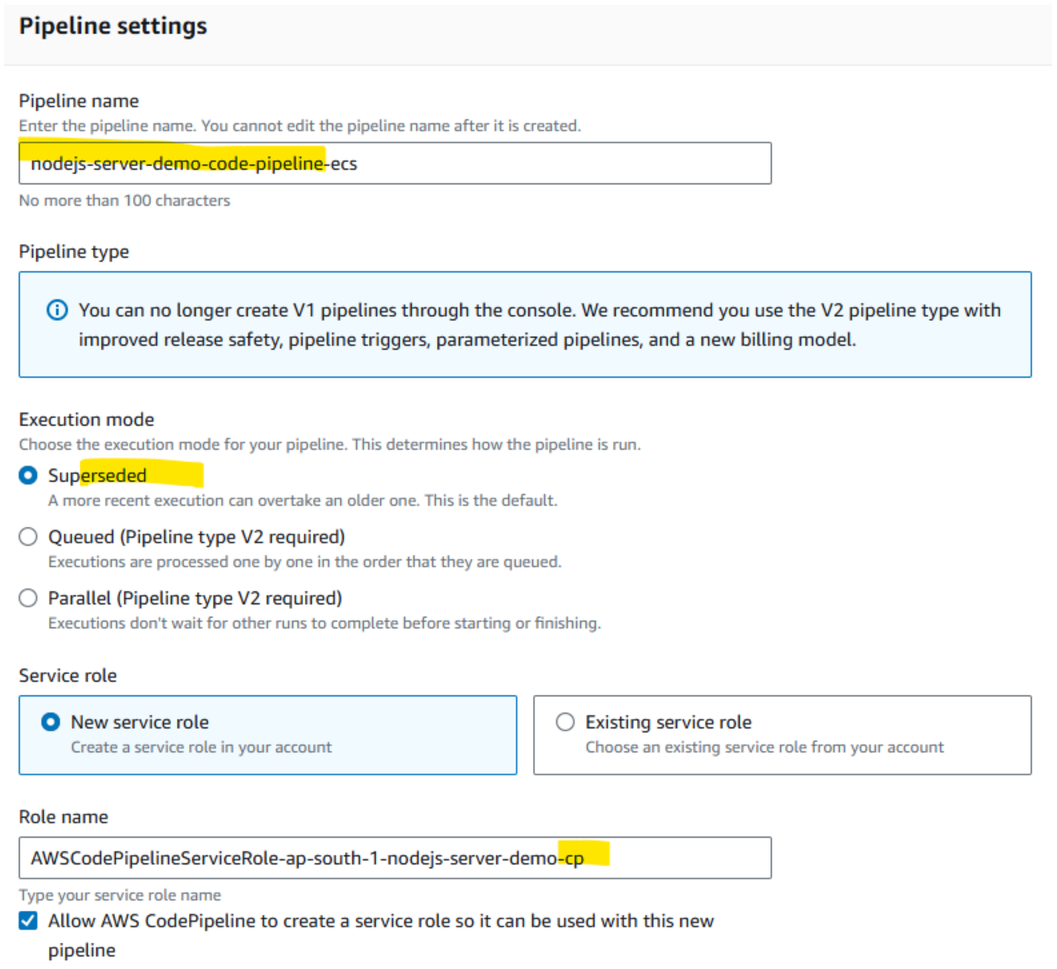
Create Code Pipeline (CI/CD):

Create Pipeline


Developer Tools > CodePipeline > Pipelines

Pipelines Info

Name	Latest execution status	Latest source revisions	Latest execution started	Most recent executions



Pipeline settings

Pipeline name

Enter the pipeline name. You cannot edit the pipeline name after it is created.

nodejs-server-demo-code-pipeline-ecs

No more than 100 characters

Pipeline type

ⓘ You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.

Execution mode

Choose the execution mode for your pipeline. This determines how the pipeline is run.

● Superseded
A more recent execution can overtake an older one. This is the default.

○ Queued (Pipeline type V2 required)
Executions are processed one by one in the order that they are queued.

○ Parallel (Pipeline type V2 required)
Executions don't wait for other runs to complete before starting or finishing.

Service role

● New service role
Create a service role in your account

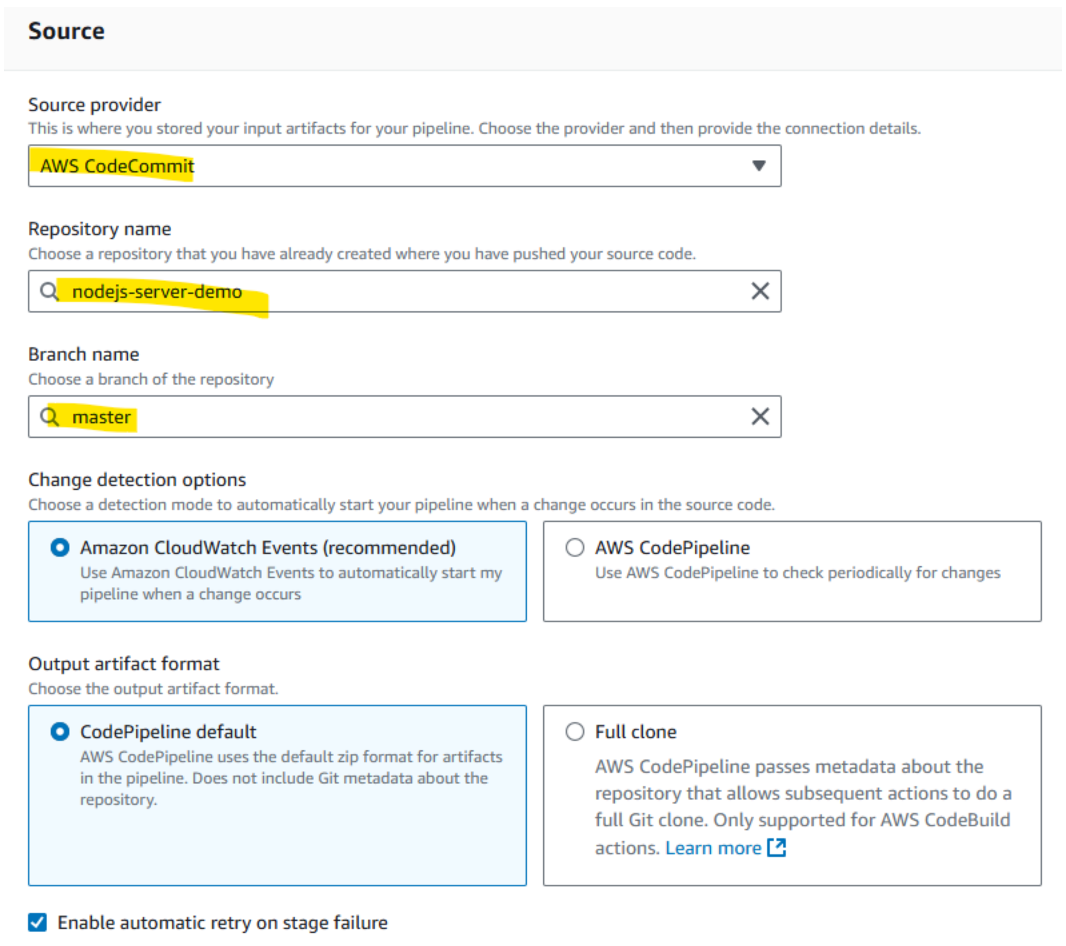
○ Existing service role
Choose an existing service role from your account

Role name

AWSCodePipelineServiceRole-ap-south-1-nodejs-server-demo-cp

Type your service role name

✓ Allow AWS CodePipeline to create a service role so it can be used with this new pipeline



Source

Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

AWS CodeCommit

Repository name
Choose a repository that you have already created where you have pushed your source code.

nodejs-server-demo

Branch name
Choose a branch of the repository

master

Change detection options
Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

[✓ Amazon CloudWatch Events (recommended)] Use Amazon CloudWatch Events to automatically start my pipeline when a change occurs

AWS CodePipeline: Use AWS CodePipeline to check periodically for changes

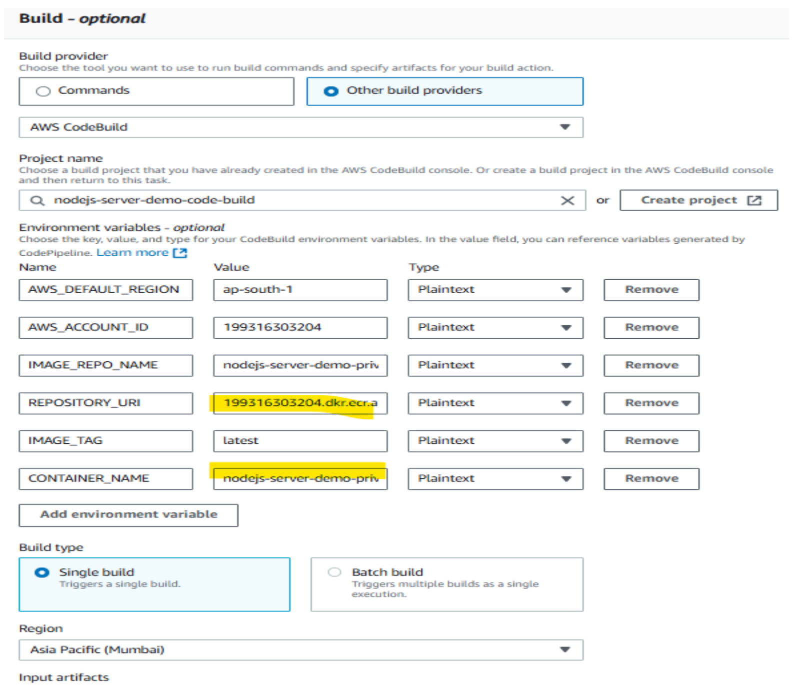
Output artifact format
Choose the output artifact format.

[✓ CodePipeline default] AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.

Full clone: AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions. Learn more

[✓] Enable automatic retry on stage failure

The container name find to ECS service task go and find it.



Build - *optional*

Build provider
Choose the tool you want to use to run build commands and specify artifacts for your build action.

Commands [✓ Other build providers]

AWS CodeBuild

Project name
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.

nodejs-server-demo-code-build or Create project

Environment variables - *optional*
Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. Learn more

Name	Value	Type	
AWS_DEFAULT_REGION	ap-south-1	Plaintext	Remove
AWS_ACCOUNT_ID	199316303204	Plaintext	Remove
IMAGE_REPO_NAME	nodejs-server-demo-priv	Plaintext	Remove
REPOSITORY_URI	199316303204.dkr.ecr.a	Plaintext	Remove
IMAGE_TAG	latest	Plaintext	Remove
CONTAINER_NAME	nodejs-server-demo-priv	Plaintext	Remove

Add environment variable

Build type

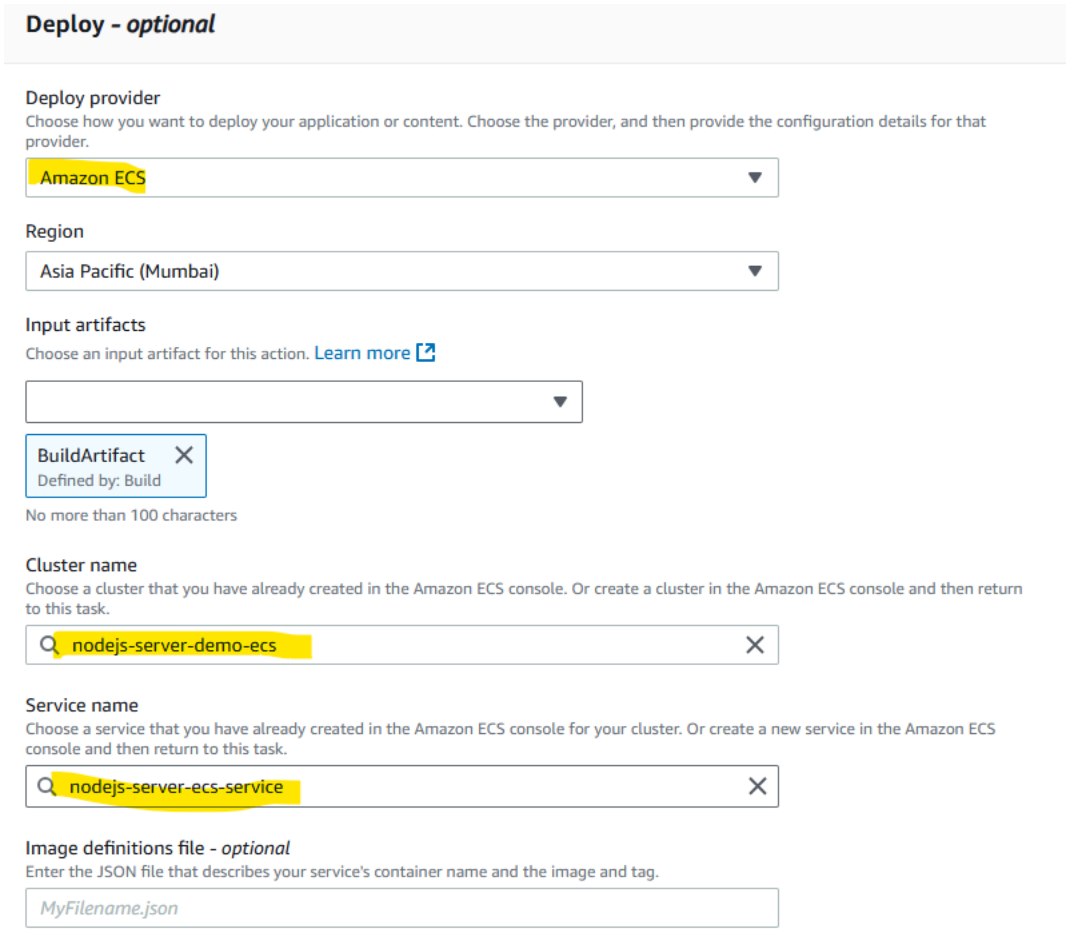
[✓ Single build] Triggers a single build.

Batch build: Triggers multiple builds as a single execution.

Region

Asia Pacific (Mumbai)

Input artifacts



Deploy - *optional*

Deploy provider
Choose how you want to deploy your application or content. Choose the provider, and then provide the configuration details for that provider.
Amazon ECS

Region
Asia Pacific (Mumbai)

Input artifacts
Choose an input artifact for this action. Learn more

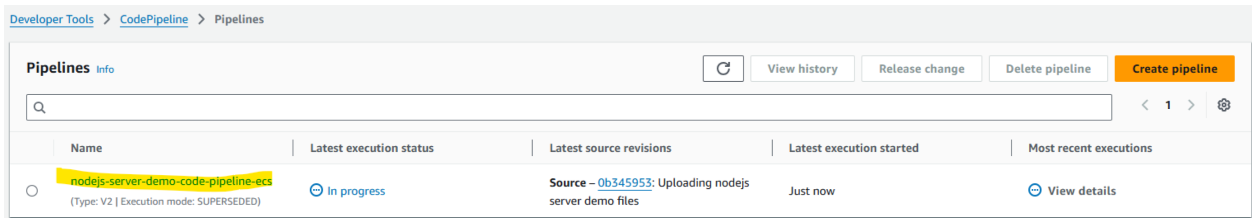
BuildArtifact
Defined by: Build

No more than 100 characters

Cluster name
Choose a cluster that you have already created in the Amazon ECS console. Or create a cluster in the Amazon ECS console and then return to this task.
nodejs-server-demo-ecs

Service name
Choose a service that you have already created in the Amazon ECS console for your cluster. Or create a new service in the Amazon ECS console and then return to this task.
nodejs-server-ecs-service

Image definitions file - *optional*
Enter the JSON file that describes your service's container name and the image and tag.
MyFilename.json

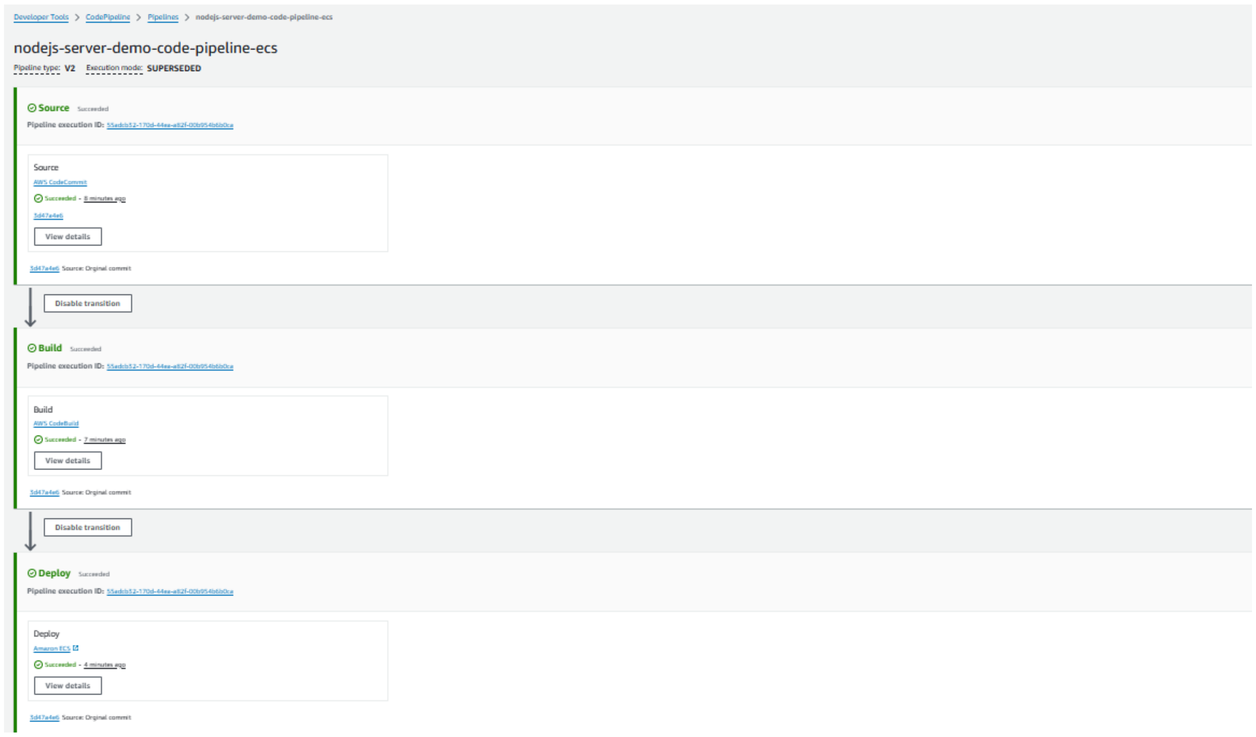


Developer Tools > CodePipeline > Pipelines

Pipelines Info

Name	Latest execution status	Latest source revisions	Latest execution started	Most recent executions
nodejs-server-demo-code-pipeline-ecs (Type: V2 | Execution mode: SUPERSEDED)	In progress	Source – 0b345953: Uploading nodejs server demo files	Just now	View details

Finally Create the Pipeline.



Developer Tools > CodePipeline > Pipelines > nodejs-server-demo-code-pipeline-ecs

nodejs-server-demo-code-pipeline-ecs

Pipeline type: V2 Execution mode: SUPERSEDED

Source Succeeded
Pipeline execution ID: 55adcb52-170d-44ea-a82f-00b954b6b0ca

Source
AWS CodeCommit
Succeeded - 8 minutes ago
3d47a4e6

3d47a4e6 Source: Orginal commit

Build Succeeded
Pipeline execution ID: 55adcb52-170d-44ea-a82f-00b954b6b0ca

Build
AWS CodeBuild
Succeeded - 7 minutes ago

3d47a4e6 Source: Orginal commit

Deploy Succeeded
Pipeline execution ID: 55adcb52-170d-44ea-a82f-00b954b6b0ca

Deploy
Amazon ECS
Succeeded - 4 minutes ago

3d47a4e6 Source: Orginal commit

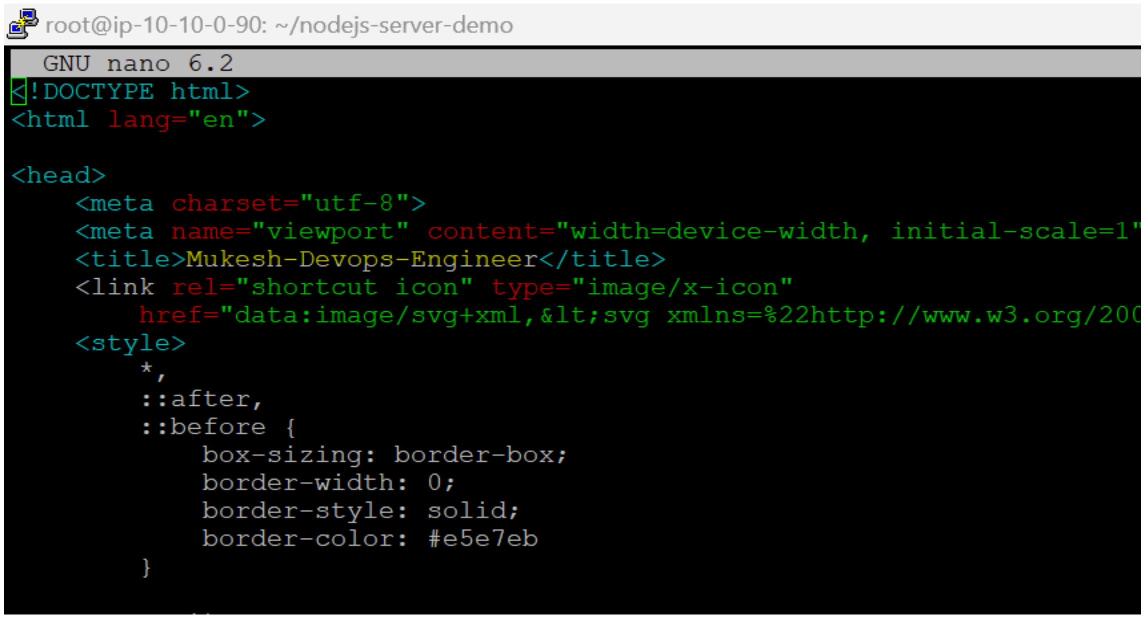
Pipeline is Successfully Completed.

CICD PIPELINE TO AWS ECS-DEMO

Go to the ssh console.

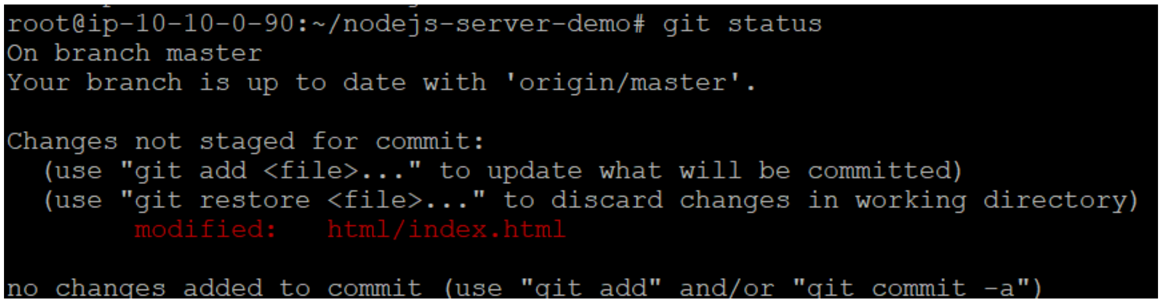
nano html/index.html

modify the html code.



```
root@ip-10-10-0-90: ~/nodejs-server-demo
  GNU nano 6.2
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1"
    <title>Mukesh-Devops-Engineer</title>
    <link rel="shortcut icon" type="image/x-icon"
        href="data:image/svg+xml,&lt;svg xmlns=%22http://www.w3.org/200
    <style>
        *,
        ::after,
        ::before {
            box-sizing: border-box;
            border-width: 0;
            border-style: solid;
            border-color: #e5e7eb
        }
```

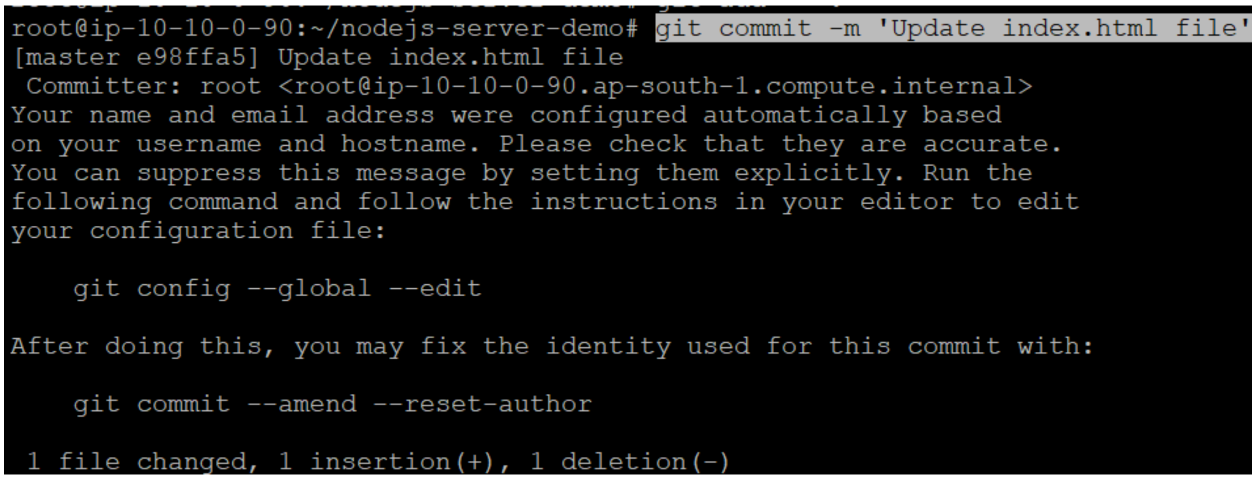
git status



```
root@ip-10-10-0-90:~/nodejs-server-demo# git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   html/index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

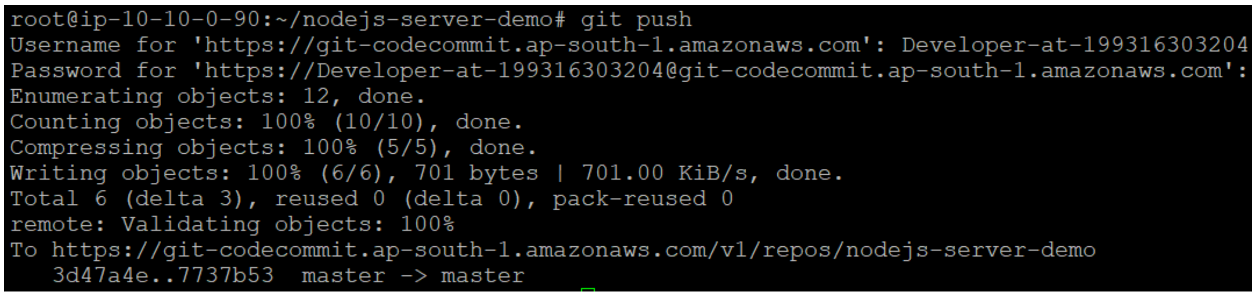
git add -- .

git commit -m 'Update index.html file'



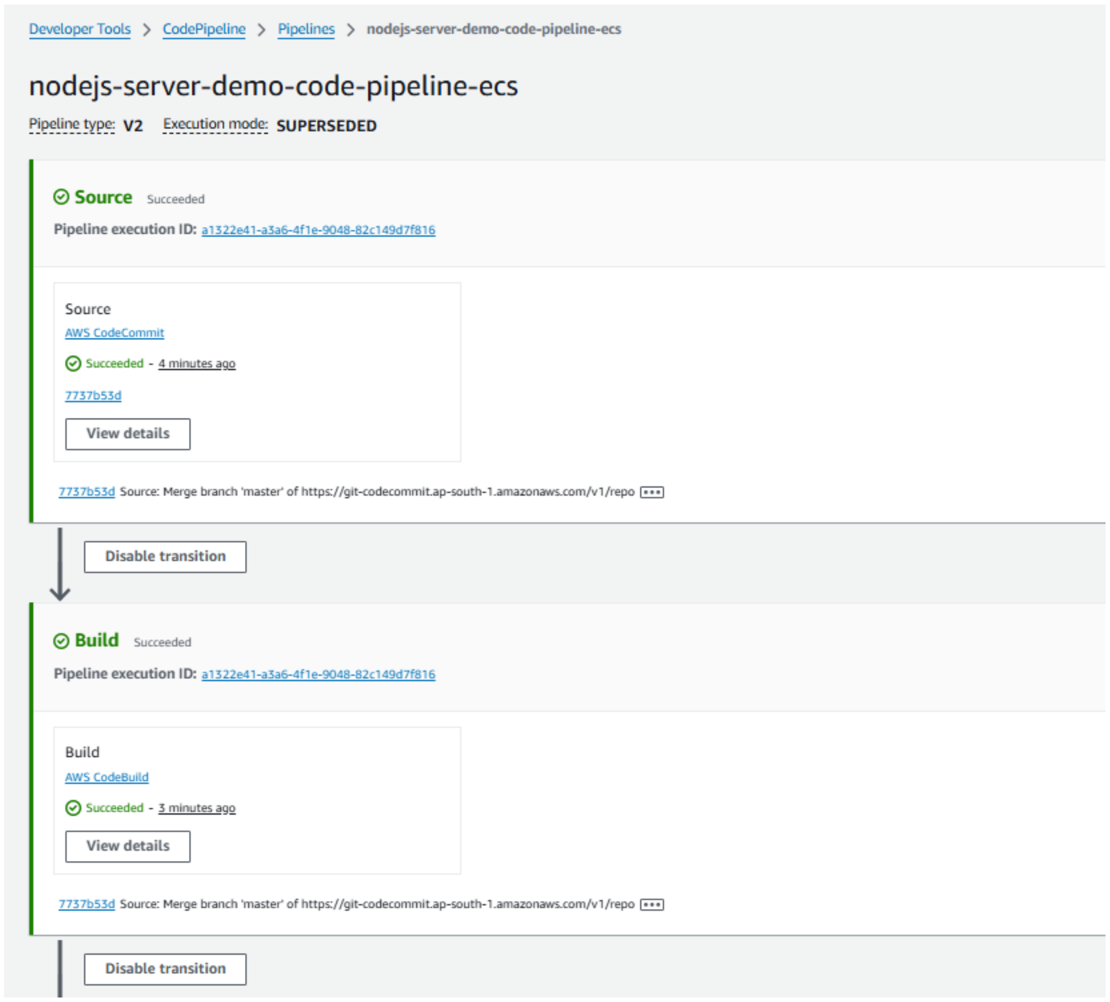
```
root@ip-10-10-0-90:~/nodejs-server-demo# git commit -m 'Update index.html file'
[master e98ffa5] Update index.html file
 Committer: root <root@ip-10-10-0-90.ap-south-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+), 1 deletion(-)
```

git push



```
root@ip-10-10-0-90:~/nodejs-server-demo# git push
Username for 'https://git-codecommit.ap-south-1.amazonaws.com': Developer-at-199316303204
Password for 'https://Developer-at-199316303204@git-codecommit.ap-south-1.amazonaws.com':
Enumerating objects: 12, done.
Counting objects: 100% (10/10), done.
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 701 bytes | 701.00 KiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
To https://git-codecommit.ap-south-1.amazonaws.com/v1/repos/nodejs-server-demo
   3d47a4e..7737b53  master -> master
```

Pipeline is Successfully Completed.



Developer Tools > CodePipeline > Pipelines > nodejs-server-demo-code-pipeline-ecs

nodejs-server-demo-code-pipeline-ecs

Pipeline type: **V2** Execution mode: **SUPERSEDED**

Source Succeeded

Pipeline execution ID: a1322e41-a3a6-4f1e-9048-82c149d7f816

Source
AWS CodeCommit
Succeeded - 4 minutes ago
7737b53d

7737b53d Source: Merge branch 'master' of https://git-codecommit.ap-south-1.amazonaws.com/v1/repo

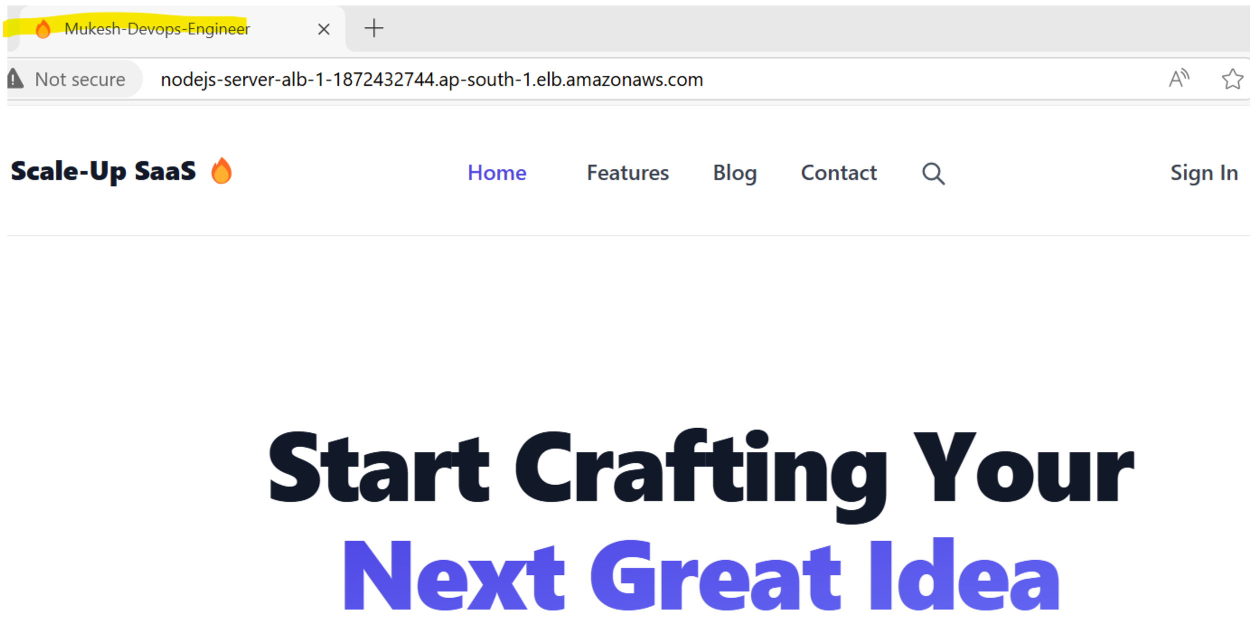
Build Succeeded

Pipeline execution ID: a1322e41-a3a6-4f1e-9048-82c149d7f816

Build
AWS CodeBuild
Succeeded - 3 minutes ago

7737b53d Source: Merge branch 'master' of https://git-codecommit.ap-south-1.amazonaws.com/v1/repo

Finally Output is.



Mukesh-Devops-Engineer

Not secure nodejs-server-alb-1-1872432744.ap-south-1.elb.amazonaws.com

Start Crafting Your
Next Great Idea