

SUPER MARIO GAME DEPLOYMENT

(AWS DEVOPS PROJECT)

Prerequisite →

1. We will use AWS account .
2. Then we configure Terraform inside an ec2 instance
3. we also need an IAM ec2 role that provide necessary permission to ec2

Completion steps →

Step 1 → Login and basics setup

Step 2 → Setup Docker ,Terraform ,aws cli , and Kubectl

Step 3 → IAM Role for EC2

Step 4 →Attach IAM role with your EC2

Step 5 → Building Infrastructure Using terraform

Step 6 → Creation of deployment and service for EKS

Step 7 → Destroy all the Insrastructure

Steps:

Step 1 : First we will Launch EC2 machine

Go to aws account

Go to Instances ->Launch instance

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get up and running by following the simple steps below.

Name and tags [Info](#)

Name

supermario-dev

[Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)


Use AMI as ubuntu


An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below


 Search our full catalog including 1000s of application and OS images


Recents


Quick Start



aws


Mac


ubuntu


Microsoft


Red Hat


[Browse more AMIs](#)
Including AMIs from
AWS, Marketplace and
the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type

Free tier eligible

ami-0866a3c8686eaebea (64-bit (x86)) / ami-0325498274077fac5 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Instance type : c7i-flex.large

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t3.medium

Family: t3 2 vCPU 4 GiB Memory Current generation: true
On-Demand SUSE base pricing: 0.0979 USD per Hour
On-Demand Windows base pricing: 0.06 USD per Hour
On-Demand Linux base pricing: 0.0416 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0451 USD per Hour
On-Demand RHEL base pricing: 0.0704 USD per Hour

☒ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

Create a new key pair

arch

[Alt+S]

N. Virginia ▼

Create key pair

×

Key pair name

Key pairs allow you to connect to your instance securely.

supermariogamekey

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

☒ RSA
RSA encrypted private and public key pair

☐ ED25519
ED25519 encrypted private and public key pair

Private key file format

☒ .pem
For use with OpenSSH

☐ .ppk
For use with PuTTY

Cancel

Create key pair

In security group allow ssh and http traffic

Additional charges apply when outside of free tier allowance

Firewall (security groups) | [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

We'll create a new security group called 'launch-wizard-17' with the following rules:

- ☒ Allow SSH traffic from Anywhere
0.0.0.0/0
Helps you connect to your instance
- ☒ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
- ☒ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

Go ahead and launch the instance

<input type="checkbox"/>	supermario-dev	i-0ed359058e9df854c	Running		t3.medium	Initializing
--------------------------	----------------	---------------------	----------------------	--	-----------	--------------

Select an instance

2) Now we will connect to the machine . For that we will first select the machine

And then click on connect button

Click on Ec2 instance connect tab

And click on connect button

Connection Type

☒ **Connect using EC2 Instance Connect**
 Connect using the EC2 Instance Connect browser-based client, with a public IPv4 or IPv6 address.

☐ **Connect using EC2 Instance Connect Endpoint**
 Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

☒ Public IPv4 address
 98.84.32.196

☐ IPv6 address
 -

Username
 Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ubuntu.

ubuntu

Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Cancel **Connect**

3) Run the following commands

- a. `sudo su`
- b. `apt update`

4) **Setup Docker ,Terraform ,aws cli , and Kubectl**

1. `apt install docker.io`

```
root@ip-172-31-35-40:/home/ubuntu# docker --version
Docker version 24.0.7, build 24.0.7-0ubuntu4.1
```

2. `usermod -aG docker $USER` # Replace with your username e.g 'ubuntu' `usermod -aG docker ubuntu`

3. `newgrp docker`

5) We will now setup terraform in the machine

```
wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
```

```
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
```

```
sudo apt update && sudo apt install terraform
```

6) Now we will setup the AWS CL

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

```
apt install unzip -y
```

```
unzip awscliv2.zip
```

```
sudo ./aws/install
```

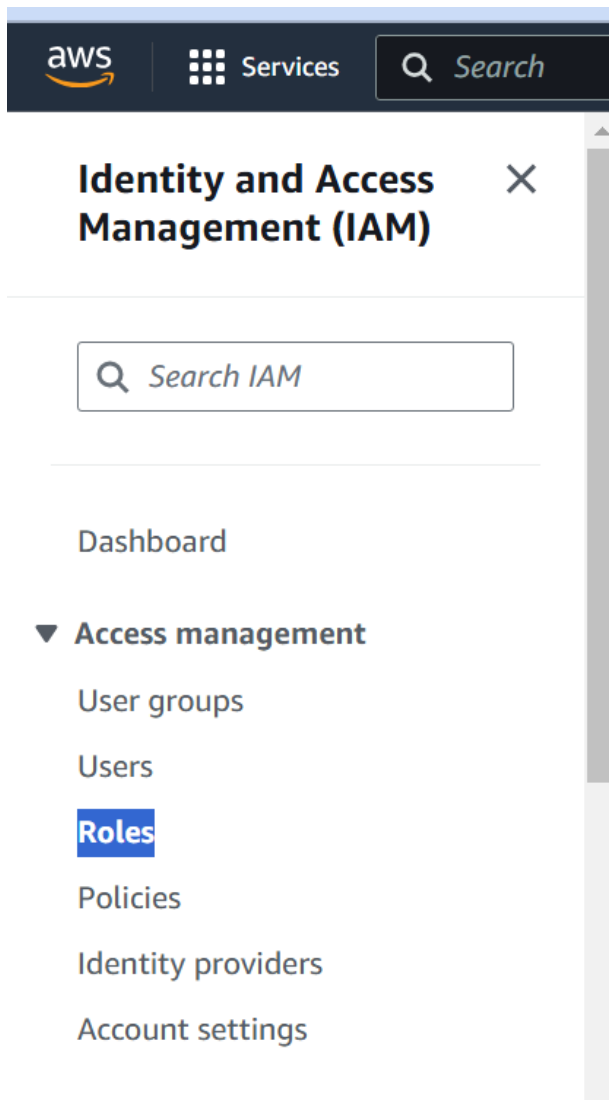
7) Now we will setup Kubectl

1. `sudo apt install curl -y`
2. `curl -LO https://dl.k8s.io/release/\$\(curl -L -s https://dl.k8s.io/release/stable.txt\)/bin/linux/amd64/kubectl`
3. `sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl`

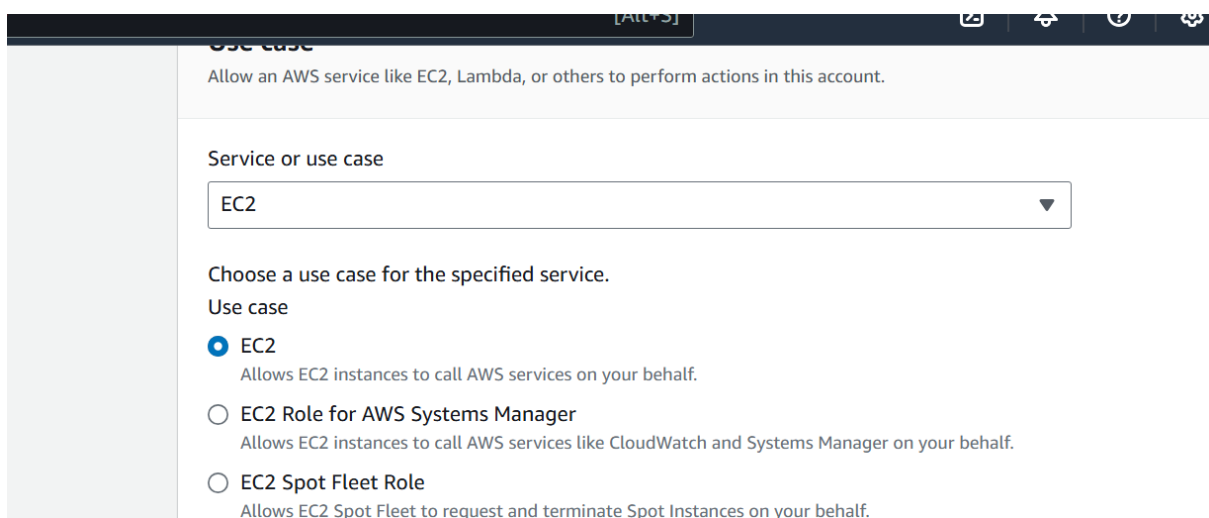
```
root@ip-172-31-35-40:/home/ubuntu# kubectl version --client
Client Version: v1.31.2
Kustomize Version: v5.4.2
```

8) Every thing is setup and installed let's make a IAM EC2 Role

Go to IAM -> Roles



click on create role and choose EC2 from the dropdown



Choose administrator access

Step 2
Add permissions

Step 3
Name, review, and create

Permissions policies (1/1022) [Info](#)








Choose one or more policies to attach to your new role.

Filter by Type

All types

< 1 2 3 4 5 6 7 ... 52 >

⚙️

<input type="checkbox"/>	Policy name ↗	Type	Description
<input checked="" type="checkbox"/>	 AdministratorAccess	AWS managed - job function	Provides full access to AWS services an...
<input type="checkbox"/>	 AdministratorAcce...	AWS managed	Grants account administrative permisi...
<input type="checkbox"/>	 AdministratorAcce...	AWS managed	Grants account administrative permisi...
<input type="checkbox"/>	 AlexaForBusinessD...	AWS managed	Provide device setup access to AlexaFo...
<input type="checkbox"/>	 AlexaForBusinessF...	AWS managed	Grants full access to AlexaForBusiness ...
<input type="checkbox"/>	 AlexaForBusinessG...	AWS managed	Provide gateway execution access to A...
<input type="checkbox"/>	 AlexaForBusinessLi...	AWS managed	Provide access to Lifesize AVS devices

Step 2

[Add permissions](#)

Step 3

Name, review, and create

Role details

Role name

Enter a meaningful name to identify this role.

Maximum 64 characters. Use alphanumeric and '+=,._-' characters.

Description

Add a short explanation for this role.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,._-/\[\]!#\$%^&*()~'"`

Step 1: Select trusted entities

Ed

Trust policy

Permissions policy summary

Policy name	Type	Attached as
AdministratorAccess	AWS managed - job function	Permissions policy

Step 3: Add tags

Add tags - optional [Info](#)

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel

Previous

Create role

9) Now we will attach the IAM role to Ec2 machine

Go to ec2 machine and select the machine

supermario-dev

i-0ed359058e9df854c

Running

t3.medium

Initializing

i-0ed359058e9df854c (supermario-dev)

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

▼ Instance summary [Info](#)

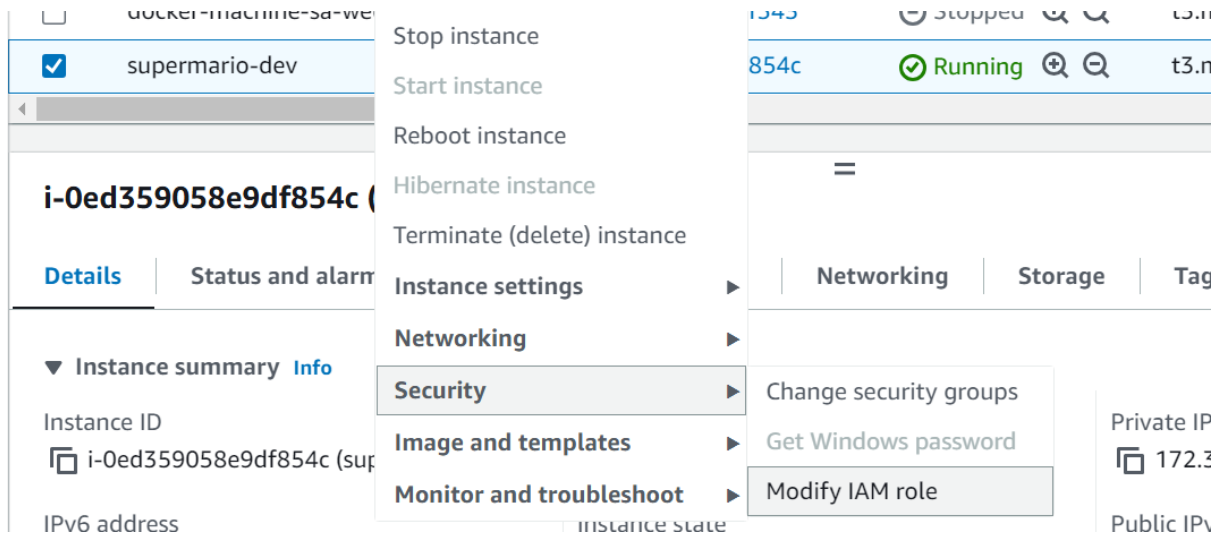
Instance ID

Public IPv4 address

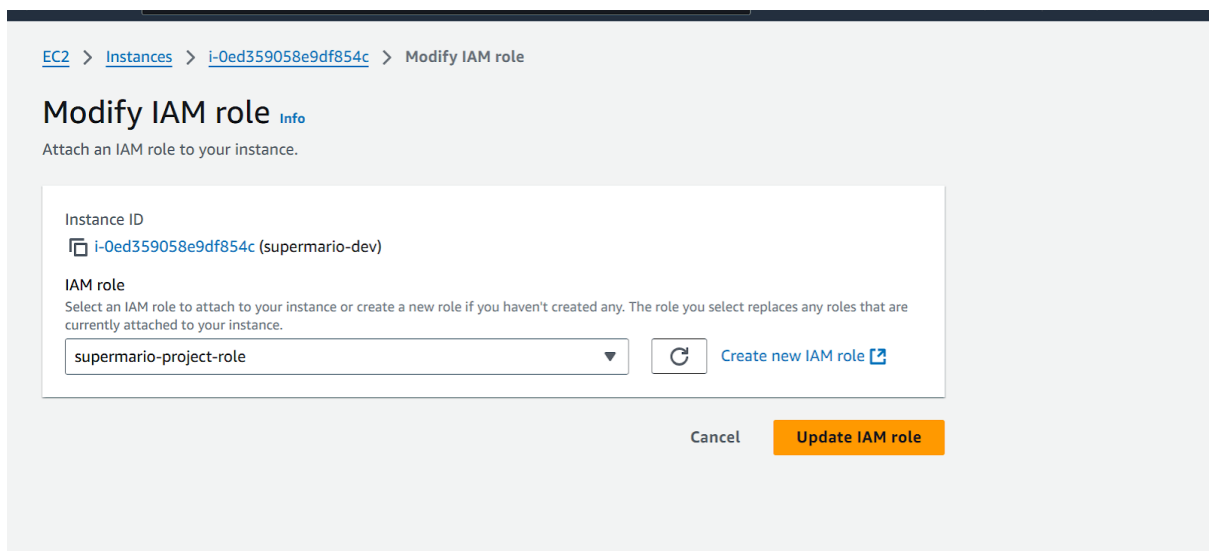
Private IPv4 addresses

Right click on the machine

Click on Security -> modify IAM role



Select IAM role which we created previously



Click on update IAM role

10) Building Infrastructure Using terraform

First we will clone the github repo

Go to the machine

```
mkdir supermario
```

```
cd supermario
```

```

customize version: v3.4.2
root@ip-172-31-35-40:/home/ubuntu# mkdir supermario
root@ip-172-31-35-40:/home/ubuntu# cd supermario
root@ip-172-31-35-40:/home/ubuntu/supermario# pwd
/home/ubuntu/supermario
root@ip-172-31-35-40:/home/ubuntu/supermario#

```

i-0ed359058e9df854c (supermario-dev)

git clone <https://github.com/akshu20791/supermario-game>

```

root@ip-172-31-35-40:/home/ubuntu/supermario# git clone https://github.com/akshu20791/supermario-game
Cloning into 'supermario-game'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 8 (delta 0), reused 8 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (8/8), done.
root@ip-172-31-35-40:/home/ubuntu/supermario#

```

i-0ed359058e9df854c (supermario-dev)

```

root@ip-172-31-35-40:/home/ubuntu/supermario# ls
supermario-game
root@ip-172-31-35-40:/home/ubuntu/supermario#

```

cd supermario-game

```

ip-172-31-35-40:/home/ubuntu/supermario# cd supermario-game
ip-172-31-35-40:/home/ubuntu/supermario/supermario-game#

```

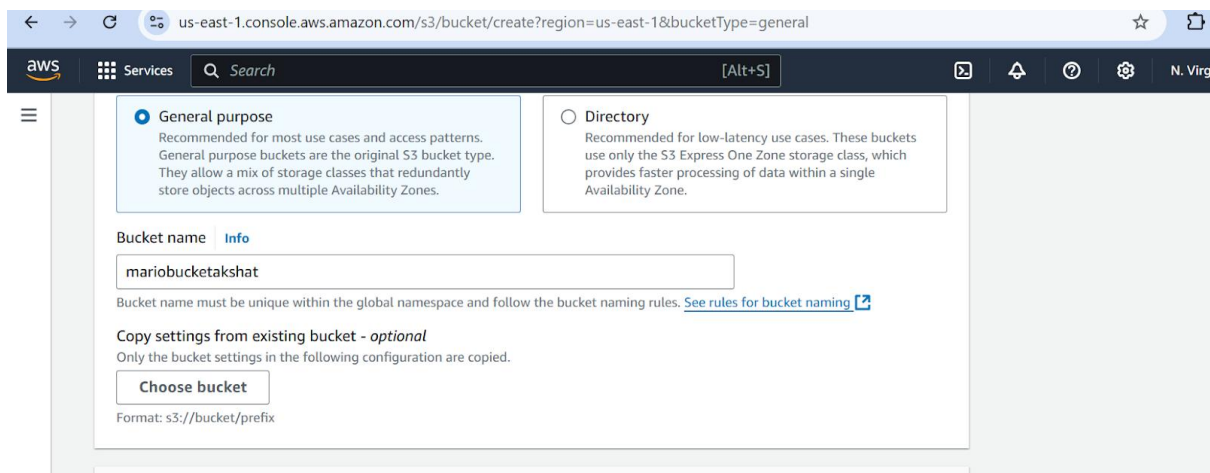
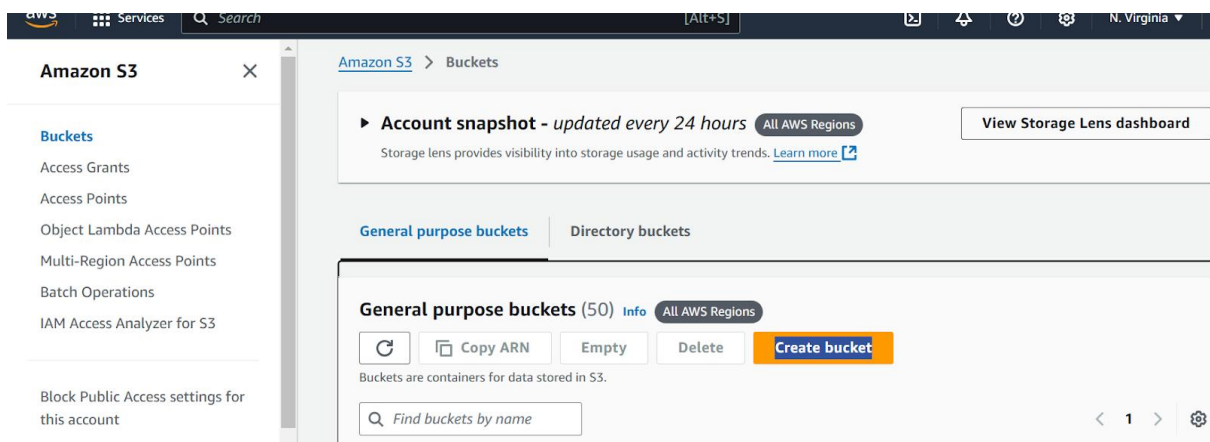
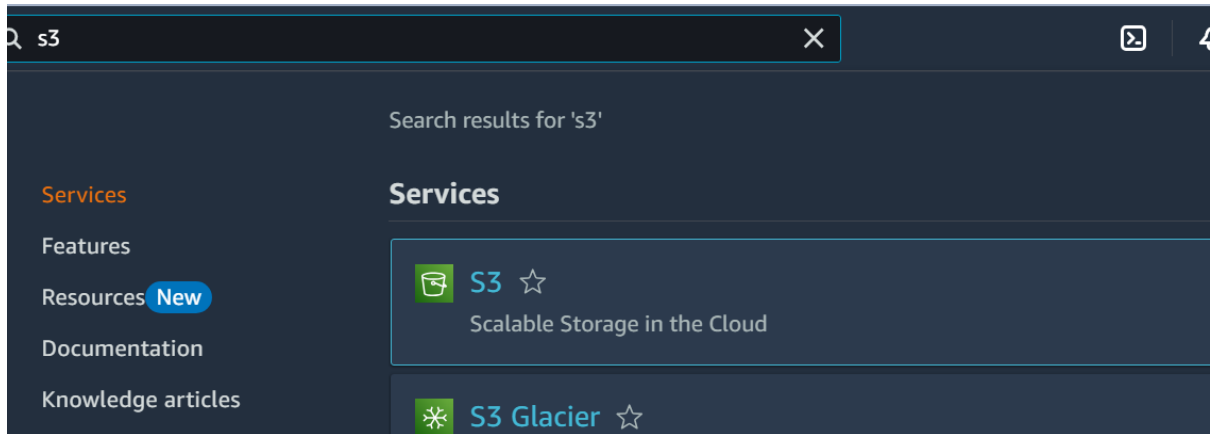
cd EKS-TF

```

root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game# cd EKS-TF
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF# ls
backend.tf  main.tf  provider.tf
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF#

```

12) Go to aws -> S3 -> Create a s3 bucket with some unique name



Create bucket

13) Go back to Ec2 machine

edit the backend.tf file by → **vim backend.tf**

```

terraform {
  backend "s3" {
    bucket = "mariobucketakshat" # Replace with your actual S3 bucket name
    key    = "EKS/terraform.tfstate"
    region = "us-east-1"
  }
}

```

Note → make sure to provide your bucket and region name in this file otherwise it doesn't work and IAM role is also associated with your ec2 which helps ec2 to use other services such S3 bucket

>> nano main.tf

Change instance type → c7i-flex.large

14) Now we will run

terraform init

```

root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF# terraform init
Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Installing hashicorp/aws v5.75.1...
- Installed hashicorp/aws v5.75.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

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

```

terraform validate

```

commands will detect it and remind you to do so if necessary.
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF# terraform validate
Success! The configuration is valid.

```

go to AWS ACCOUNT -> GO TO VPC -> GO TO SUBNETS -> DELETE subnet in availability zone us-east-1e

terraform plan

```
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF# terraform plan
data.aws_vpc.default: Reading...
data.aws_iam_policy_document.assume_role: Reading...
data.aws_iam_policy_document.assume_role: Read complete after 0s [id=3552664922]
data.aws_vpc.default: Read complete after 2s [id=vpc-0b918116e453a16a9]
data.aws_subnets.public: Reading...
data.aws_subnets.public: Read complete after 0s [id=ap-south-1]

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

Terraform will perform the following actions:

# aws_eks_cluster.example will be created
```

```
Terraform will perform the following actions:

# aws_eks_cluster.example will be created
+ resource "aws_eks_cluster" "example" {
  + arn = (known after apply)
  + bootstrap_self_managed_addons = true
  + certificate_authority = (known after apply)
  + cluster_id = (known after apply)
  + created_at = (known after apply)
  + endpoint = (known after apply)
  + id = (known after apply)
  + identity = (known after apply)
  + name = "EKS_CLOUD"
  + platform_version = (known after apply)
  + role_arn = (known after apply)
  + status = (known after apply)
  + tags_all = (known after apply)
  + version = (known after apply)
```

terraform apply --auto-approve

```
Plan: 8 to add, 0 to change, 0 to destroy.
aws_iam_role.example: Creating...
aws_iam_role.example1: Creating...
aws_iam_role.example: Creation complete after 0s [id=eks-cluster-cloud]
aws_iam_role_policy_attachment.example-AmazonEKSClusterPolicy: Creating...
aws_iam_role.example1: Creation complete after 0s [id=eks-node-group-cloud]
aws_iam_role_policy_attachment.example-AmazonEKSWorkerNodePolicy: Creating...
aws_iam_role_policy_attachment.example-AmazonEKS_CNI_Policy: Creating...
aws_iam_role_policy_attachment.example-AmazonEC2ContainerRegistryReadOnly: Creating...
aws_iam_role_policy_attachment.example-AmazonEKSClusterPolicy: Creation complete after 0s [id=eks-cluster-cloud-20240114062220905000000001]
aws_eks_cluster.example: Creating...
aws_iam_role_policy_attachment.example-AmazonEC2ContainerRegistryReadOnly: Creation complete after 0s [id=eks-node-group-cloud-202401140622209473000000002]
aws_iam_role_policy_attachment.example-AmazonEKS_CNI_Policy: Creation complete after 0s [id=eks-node-group-cloud-202401140622209789000000003]
aws_iam_role_policy_attachment.example-AmazonEKSWorkerNodePolicy: Creation complete after 0s [id=eks-node-group-cloud-202401140622209862000000004]
aws_eks_cluster.example: Still creating... [10s elapsed]
aws_eks_cluster.example: Still creating... [20s elapsed]
aws_eks_cluster.example: Still creating... [30s elapsed]
aws_eks_cluster.example: Still creating... [40s elapsed]
aws_eks_cluster.example: Still creating... [50s elapsed]
aws_eks_cluster.example: Still creating... [1m0s elapsed]
aws_eks_cluster.example: Still creating... [1m10s elapsed]
aws_eks_cluster.example: Still creating... [1m20s elapsed]
aws_eks_cluster.example: Still creating... [1m30s elapsed]
aws_eks_cluster.example: Still creating... [1m40s elapsed]
aws_eks_cluster.example: Still creating... [1m50s elapsed]
```

It will take 10-15 minutes to complete

```
aws_eks_cluster.example: Still creating... [5m40s elapsed]
aws_eks_cluster.example: Still creating... [5m50s elapsed]
aws_eks_cluster.example: Still creating... [6m0s elapsed]
aws_eks_cluster.example: Creation complete after 6m9s [id=EKS_CLOUD]
aws_eks_node_group.example: Creating...
aws_eks_node_group.example: Still creating... [10s elapsed]
aws_eks_node_group.example: Still creating... [20s elapsed]
aws_eks_node_group.example: Still creating... [30s elapsed]
aws_eks_node_group.example: Still creating... [40s elapsed]
aws_eks_node_group.example: Still creating... [50s elapsed]
aws_eks_node_group.example: Still creating... [1m0s elapsed]
aws_eks_node_group.example: Still creating... [1m10s elapsed]
aws_eks_node_group.example: Still creating... [1m20s elapsed]
aws_eks_node_group.example: Still creating... [1m30s elapsed]
aws_eks_node_group.example: Still creating... [1m40s elapsed]
aws_eks_node_group.example: Still creating... [1m50s elapsed]
aws_eks_node_group.example: Creation complete after 1m51s [id=EKS_CLOUD:Node-cloud]
Apply complete! Resources: 8 added, 0 changed, 0 destroyed.
```

15) Now we will update the configuration of EKS

```
aws eks update-kubeconfig --name EKS_CLOUD --region us-east-1
```

ensure that you take care of the region you are working in

```
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF# aws eks update-kubeconfig --name EKS_CLOUD --region ap-south-1
Added new context arn:aws:eks:ap-south-1:975050323630:cluster/EKS_CLOUD to /root/.kube/config
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF#
```

16) Creation of deployment and service for EKS

1. change the directory where deployment and service files are stored use the command → **cd**

..

```
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game/EKS-TF# cd ..
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game# ls
EKS-TF  deployment.yaml  service.yaml
```

2. create the deployment

```
kubectl apply -f deployment.yaml
```

```
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game# kubectl apply -f deployment.yaml
deployment.apps/mario-deployment created
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game#
```

```
kubectl apply -f service.yaml
```

```
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game# kubectl apply -f service.yaml
service/mario-service created
root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game#
```

run → **kubectl get all**

```

root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game# kubectl get all
NAME                                READY    STATUS    RESTARTS   AGE
pod/mario-deployment-6bfd49df4b-tgzrd 1/1      Running   0           64s
pod/mario-deployment-6bfd49df4b-vxf7q 1/1      Running   0           64s

NAME                                TYPE                CLUSTER-IP    EXTERNAL-IP    PORT(S)
service/kubernetes                  ClusterIP          10.100.0.1     <none>          443/TCP
service/mario-service              LoadBalancer       10.100.68.49   a086c1c645bc8444f943afe7a74c47a5-1256226015.ap-south-1.elb.amazonaws.com 80:3153/TCP

NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/mario-deployment    2/2      2              2            65s

NAME                                DESIRED    CURRENT    READY    AGE
replicaset.apps/mario-deployment-6bfd49df4b 2          2          2        65s

```

i-Oed359058e9df854c (supermario-dev)

PublicIPs: 98.84.32.196 PrivateIPs: 172.31.35.40

16) Now Run the following command to get the load balancer ingress

This command tells all the details of your application

kubectl describe service mario-service

```

root@ip-172-31-35-40:/home/ubuntu/supermario/supermario-game# kubectl describe service mario-service
Name: mario-service
Namespace: default
Labels: <none>
Annotations: <none>
Selector: app=mario
Type: LoadBalancer
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.100.68.49
IPs: 10.100.68.49
LoadBalancer Ingress: a086c1c645bc8444f943afe7a74c47a5-1256226015.ap-south-1.elb.amazonaws.com
Port: <unset> 80/TCP
TargetPort: 80/TCP
NodePort: <unset> 31553/TCP
Endpoints: 172.31.8.189:80,172.31.9.189:80
Session Affinity: None
External Traffic Policy: Cluster
Internal Traffic Policy: Cluster
Events:
  Type     Reason             Age   From              Message
  ----     -
  Normal   EnsuringLoadBalancer 93s   service-controller Ensuring load balancer
  Normal   EnsuredLoadBalancer 89s   service-controller Ensured load balancer

```

Here you will see load balancer ingress link

Copy that

The screenshot shows a terminal window with the output of the command `kubectl describe service mario-service`. A context menu is open over the load balancer ingress link `a086c1c645bc8444f943afe7a74c47a5-1256226015.ap-south-1.elb.amazonaws.com`. The menu options include Cut, Copy, Paste, Paste as plain text, Select all, Open in reading mode, Go to a086c1c645bc8444f943afe7a74c47a5-1256226015.ap-south-1.elb.amazonaws.com, Print..., and Open in reading mode. The terminal output shows the service details, including the load balancer ingress link and the endpoints.

17) paste it in the browser

Ensure that when you copy paste ...it should be http not https

