Create IAM role for the EKS cluster (**AmazonEKSAutoClusterRole)**- Attach permissions policies section, search for and select the appropriate policies for your EKS cluster, **AmazonEKSClusterPolicy**, **AmazonEKSServicePolicy**  
  
Create IAM role for the EKS NodeGroup EC2 instance **(AutoEKSAutoNodeRole)** - In the **Attach permissions policies** section, search for and select the appropriate policies for your EKS cluster, **AmazonEKSWorkerNodePolicy, AmazonEKS\_CNI\_Policy, AmazonEC2ContainerRegistryReadOnly**  
  
Copy your files to you environment (local computer or ec2)  
scp -i "./key/test.pem" C:\Users\HP\Documents\cloudspeed\deepseek-project\eks\\* ec2-user@3.12.146.230:/home/ec2-user/  
  
SSH to your instance  
ssh -i "./key/test.pem" [ec2-user@3.12.146.230](mailto:ec2-user@3.12.146.230)

Install AWS CLI on your environment  
sudo yum update -y

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"  
sudo yum install -y unzip  
unzip awscliv2.zip  
sudo ./aws/install  
aws –version  
rm -rf awscliv2.zip aws/  
  
Configure your command line user

aws configure

Create your Cluster  
aws eks create-cluster --name <cluster-name> --role-arn role-arn  --resources-vpc-config "subnetIds=subnet1,subnet2,securityGroupIds=sg-1"  
  
  
  
  
Create your ECR Repository

aws ecr create-repository --repository-name ollama --region us-east-2

aws ecr create-repository --repository-name fastapi --region us-east-2   
  
- Install Docker -  
sudo yum update -y

sudo yum install -y docker  
  
sudo systemctl start docker

sudo systemctl enable docker  
  
sudo usermod -aG docker ec2-user

sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

docker-compose version  
exit (log back in)

Build your Images  
docker build -t ollama -f Dockerfile.ollama .  
docker build -t fastapi -f Dockerfile .  
  
Confirm your container files are given the necessary tag  
docker tag ollama 664418975315.dkr.ecr.us-east-2.amazonaws.com/ollama

docker tag fastapi 664418975315.dkr.ecr.us-east-2.amazonaws.com/fastapi

Confirm your Identity  
aws sts get-caller-identity  
  
Authenticate before you can push your image to ecr

aws ecr get-login-password --region us-east-2 | docker login --username AWS --password-stdin 664418975315.dkr.ecr.us-east-2.amazonaws.com

## Push your image to ecr

docker push 664418975315.dkr.ecr.us-east-2.amazonaws.com/ollama

docker push 664418975315.dkr.ecr.us-east-2.amazonaws.com/fastapi

## Verify the images are in ECR

aws ecr describe-images --repository-name ollama --region us-east-2

aws ecr describe-images --repository-name fastapi --region us-east-2  
  
Update deployment files using your image URI  
image: 664418975315.dkr.ecr.us-east-2.amazonaws.com/fastapi

image: 664418975315.dkr.ecr.us-west-2.amazonaws.com/ollama

Install KubeCTL  
curl -LO <https://dl.k8s.io/release/v1.31.0/bin/linux/amd64/kubectl>

chmod +x kubectl

sudo mv kubectl /usr/local/bin/

## Confirm cluster is running -

aws eks update-kubeconfig --region us-east-2 --name deepseek  
  
kubectl config get-contexts

else you can set the context  
kubectl config use-context arn:aws:eks:eu-west-2:664418975315:cluster/deepseek

## Create NodeGroups (at least 30GB memory)

Use instance r4.2xlarge

Get Nodes for KubeCTL

kubectl get nodes

Create Service Account for Traefik  
kubectl create serviceaccount traefik-sa

Install Traefic CRD  
kubectl apply -f <https://raw.githubusercontent.com/traefik/traefik/v2.10.4/docs/content/reference/dynamic-configuration/kubernetes-crd-definition-v1.yml>  
Verify Installation  
kubectl get crds | grep traefik  
  
Start your Application

Deploy PVC  
kubectl apply -f ollama-pvc.yaml

Deploy Ollama  
kubectl apply -f ollama-deployment.yaml

kubectl apply -f ollama-service.yaml

Deploy FatsAPI

kubectl apply -f ollama-deployment.yaml

kubectl apply -f ollama-service.yaml  
  
  
Deploy Traefik  
kubectl apply -f fastapi-ingressroute.yaml

kubectl apply -f traefik-pvc.yaml  
kubectl apply -f traefik-deployment.yaml

Or deploy everything in this order  
kubectl apply -f traefik-configmap.yaml

kubectl apply -f traefik-clusterrole.yaml  
kubectl apply -f traefik-clusterrolebinding.yaml

kubectl apply -f fastapi-service.yaml

kubectl apply -f ollama-service.yaml

kubectl apply -f traefik-service.yaml

kubectl apply -f fastapi-deployment.yaml

kubectl apply -f ollama-deployment.yaml

kubectl apply -f fastapi-ingress.yaml

kubectl apply -f fastapi-ingressroute.yaml

Get your Traefik External IP  
kubectl get svc traefik

Get the IP attached to your ELB (Traefik load balancer DNS address)

nslookup <traefik-external-DNS address>

Update your DNS provider with your ELB IP or DNS address

kubectl apply -f traefik-deployment.yaml

Confirm deployment  
  
kubectl get pods

kubectl get services

kubectl get ingress

kubectl logs deploy/traefik   
  
Confirm all the pods are running without errors  
kubectl logs -l app=ollama --tail=50

kubectl logs -l app=fastapi --tail=50

kubectl logs -l app=traefik --tail=50  
  
Restart traefik if you have errors from the previous command

kubectl rollout restart deployment traefik

Get your Traefik External IP  
kubectl get svc traefik  
  
Get the IP attached to your ELB (Traefik load balancer)

nslookup a038e60a8b1454351a525a255a24e8e0-168272704.us-east-2.elb.amazonaws.com  
  
update your domain provider with the IP address  
  
Wait for some time

## - Test your API (remember to change it to match your domain name)-

curl -X GET "https://superearner.online/"

curl -X POST "https://superearner.online/generate" -H "Content-Type: application/json" -d "{\"prompt\": \"say hi\", \"model\": \"deepseek-r1:7b\", \"stream\": false, \"max\_tokens\": 300, \"temperature\": 0.4}"

## - Start your streamlit app -

streamlit run ./app/app.py

## - Query your app -

Hi, hello etc