TASK 2

EKS deployment using code pipelines

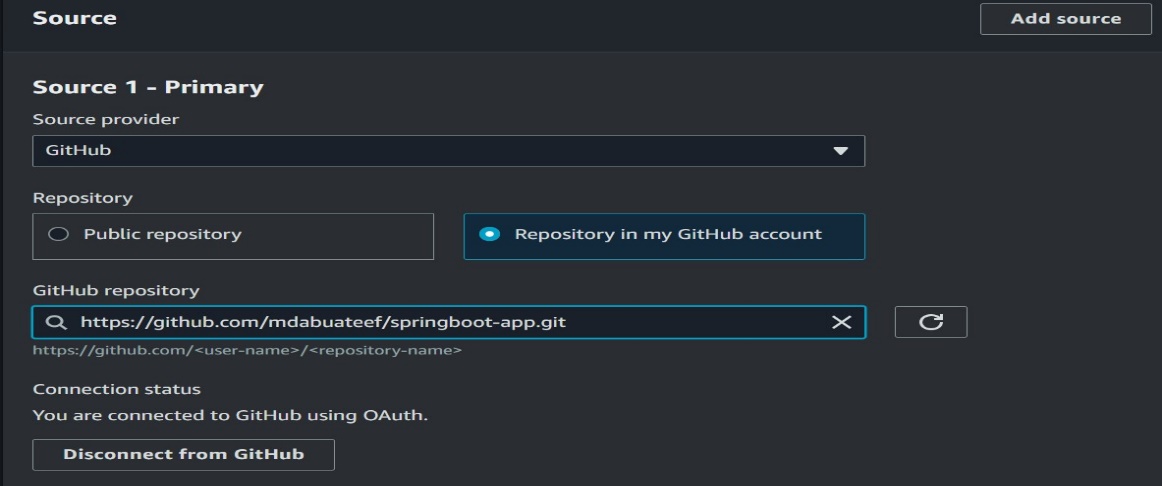
Taking any sample application and deploy in EKS using the source code as Bitbucket/GitHub

1. Open console and navigate to codebuild

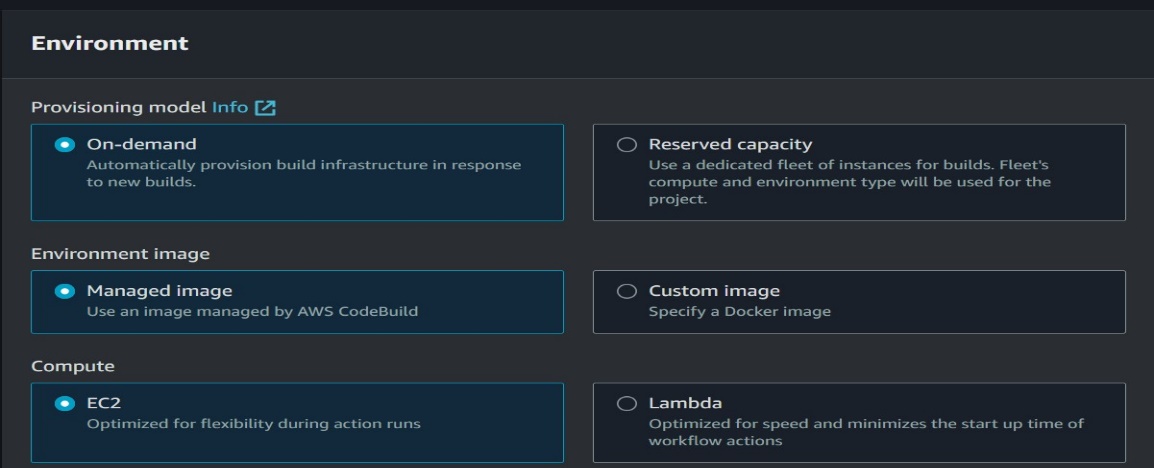
Provide a logical name for the project and select the source as git hub

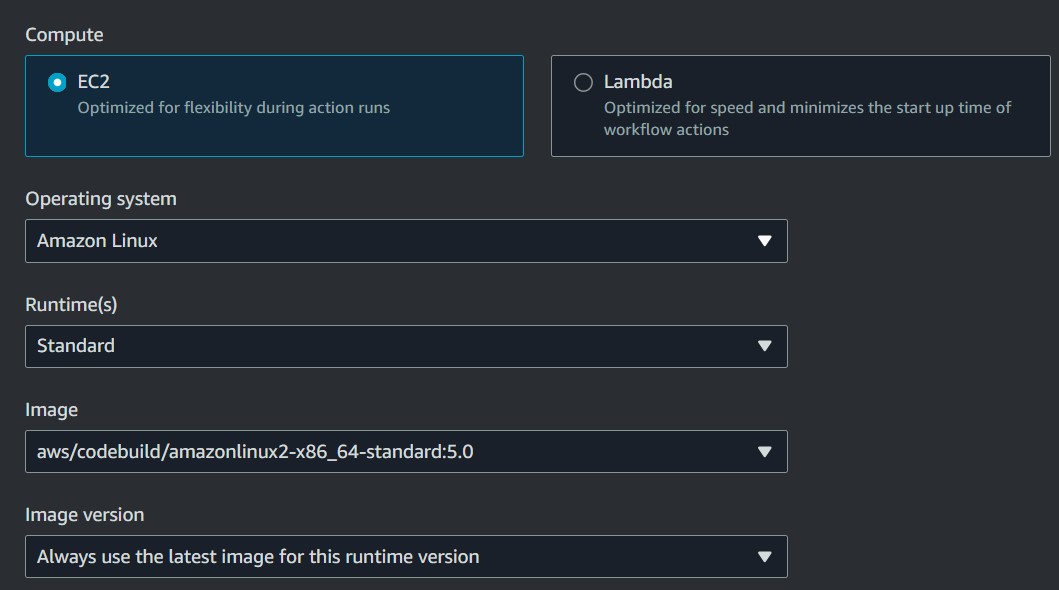
And connect with the GitHub after allowing it to access to GitHub

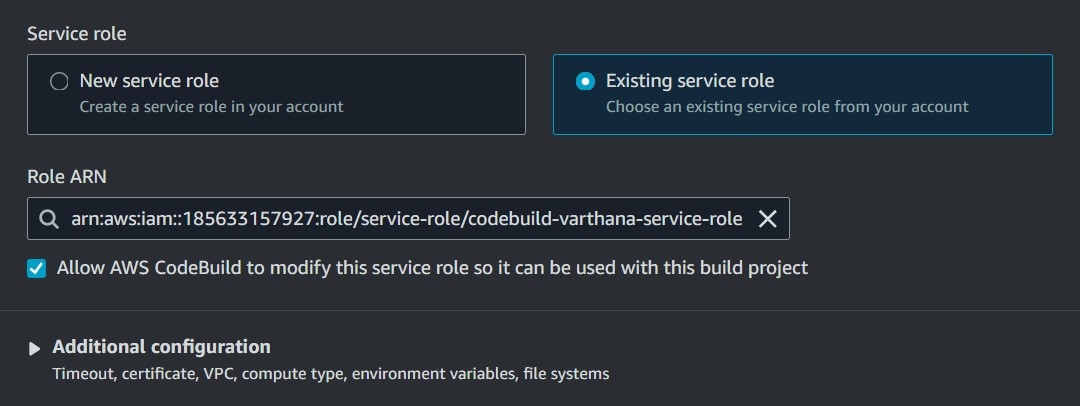
Provide the repo name and select it

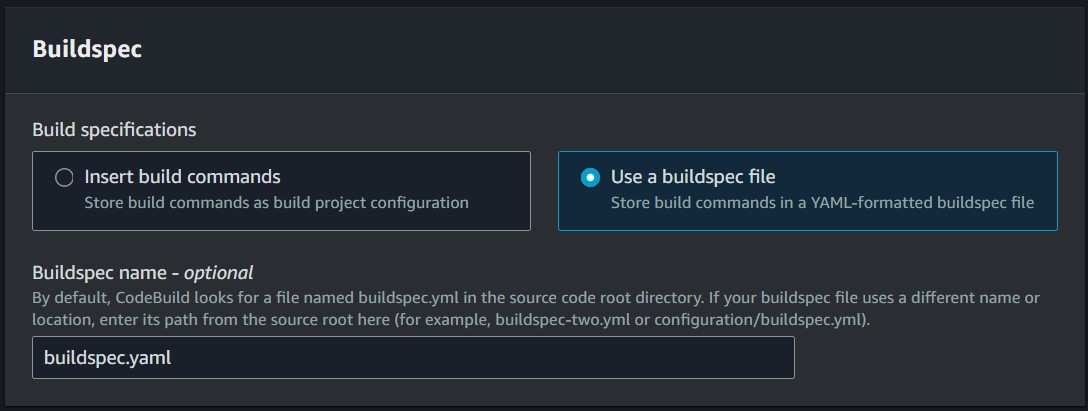


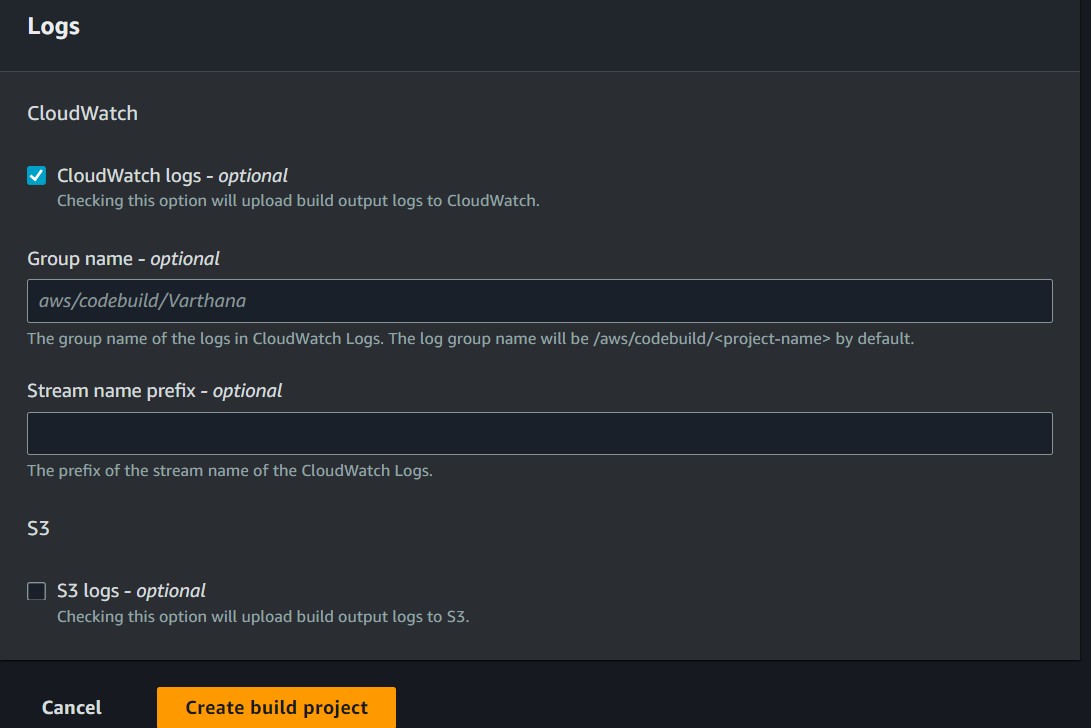
Provide the environment details to build the code





 Choose to create a new service role

 Choose buildspec.yaml file as to run

 Choose to store CloudWatch logs

1. Create a cluster by using command

Eksctl create cluster --name spring --version 1.29 --region ap-south-1 --nodegroup-name two-springs --node-type t2.large –managed

1. Create a role.sh file copy the contents into it and run the file providing +x permissions this will create a role with name EksCodeBuildKubectlRole and add an inline permission for eks describe.

# Export your Account ID

export ACCOUNT\_ID=180789647333

# Set Trust Policy

TRUST="{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Effect\": \"Allow\", \"Principal\": { \"AWS\": \"arn:aws:iam::${ACCOUNT\_ID}:root\" }, \"Action\": \"sts:AssumeRole\" } ] }"

# Verify inside Trust policy, your account id got replacd

echo $TRUST

# Create IAM Role for CodeBuild to Interact with EKS

aws iam create-role --role-name EksCodeBuildKubectlRole --assume-role-policy-document "$TRUST" --output text --query 'Role.Arn'

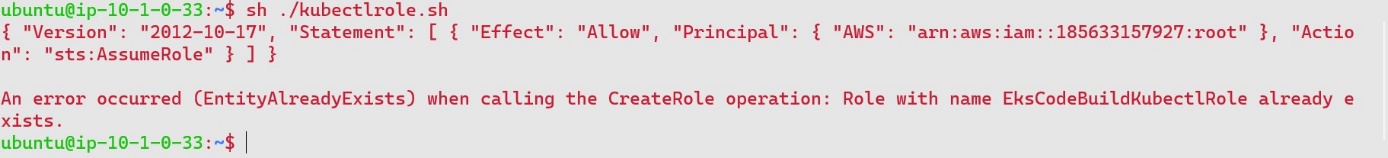
# Define Inline Policy with eks Describe permission in a file iam-eks-describe-policy

echo '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "eks:Describe\*", "Resource": "\*" } ] }' > /tmp/iam-eks-describe-policy

# Associate Inline Policy to our newly created IAM Role

aws iam put-role-policy --role-name EksCodeBuildKubectlRole --policy-name eks-describe --policy-document file:///tmp/iam-eks-describe-policy

# Verify the same on Management Console



1. Update EKS Cluster aws-auth ConfigMap with new role created by running the below awsauth.sh file

# Verify what is present in aws-auth configmap before change

kubectl get configmap aws-auth -o yaml -n kube-system

# Export your Account ID

export ACCOUNT\_ID= <your-acc-id>

# Set ROLE value

ROLE=" - rolearn: arn:aws:iam::$ACCOUNT\_ID:role/EksCodeBuildKubectlRole\n username: build\n groups:\n - system:masters"

# Get current aws-auth configMap data and attach new role info to it

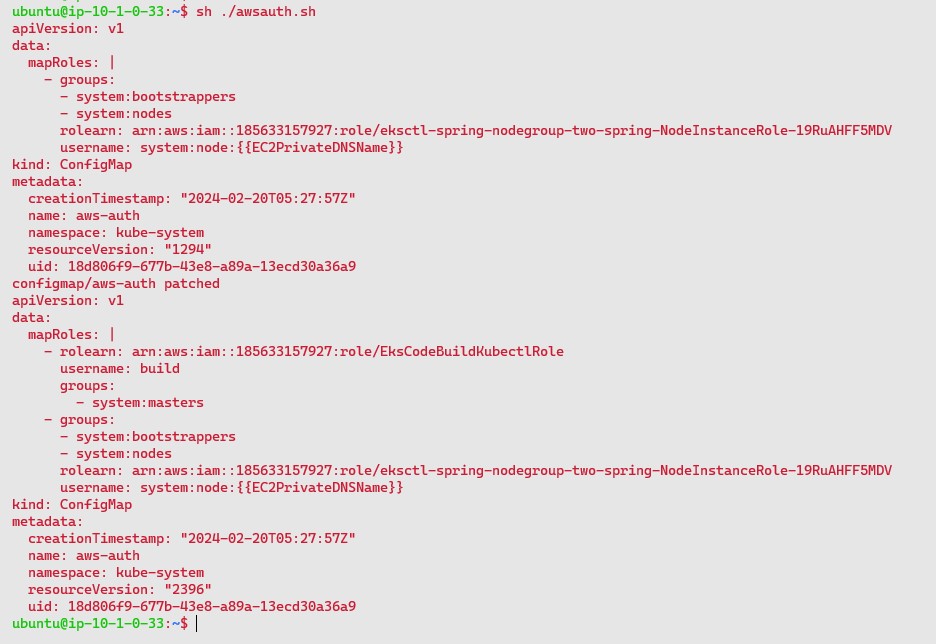
kubectl get -n kube-system configmap/aws-auth -o yaml | awk "/mapRoles: \|/{print;print \"$ROLE\";next}1" > /tmp/aws-auth-patch.yml

# Patch the aws-auth configmap with new role

kubectl patch configmap/aws-auth -n kube-system --patch "$(cat /tmp/aws-auth-patch.yml)"

# Verify what is updated in aws-auth configmap after change

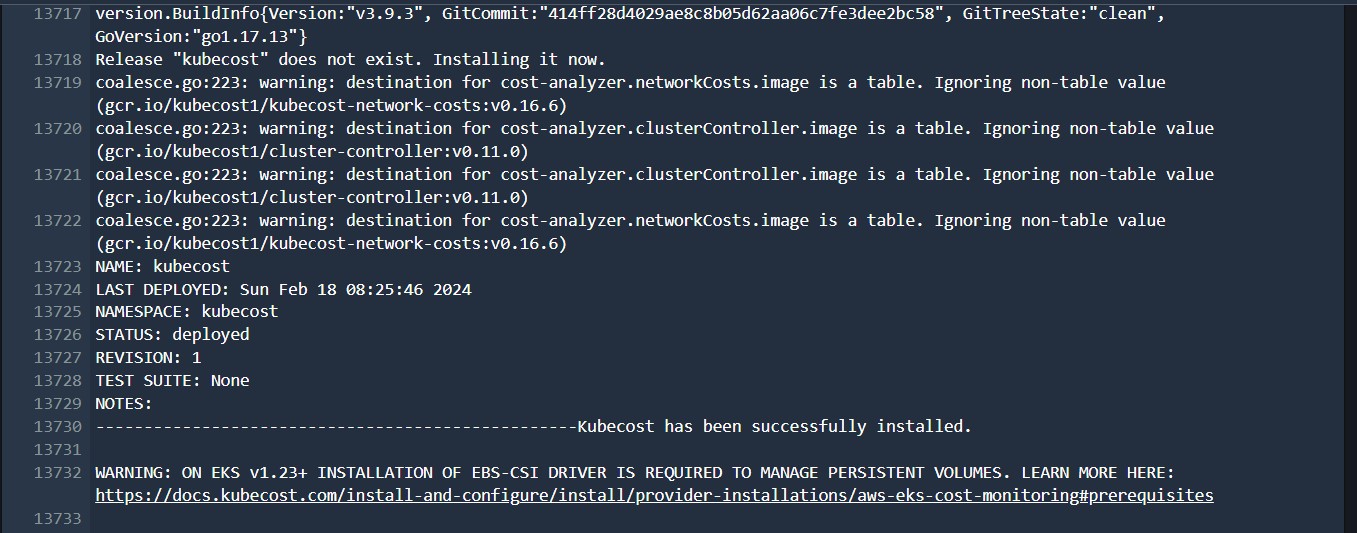
kubectl get configmap aws-auth -o yaml -n kube-system



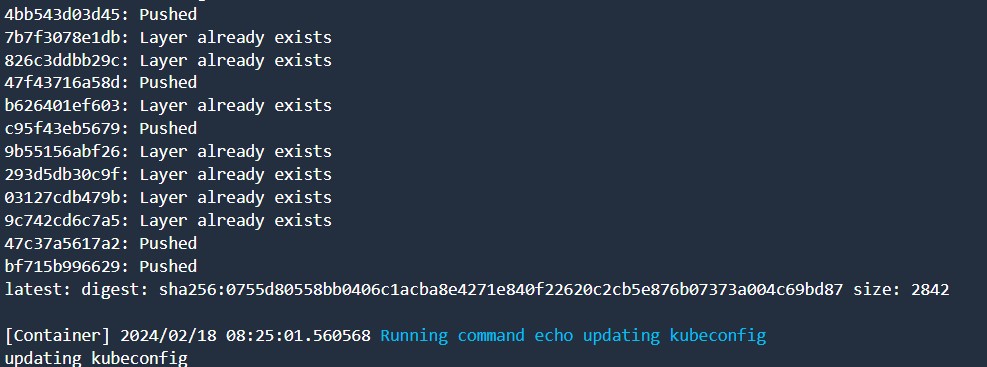
1. Create a build spec file and push it to the git hub repo with the manifest files and the required csir driver and awscli, kubectl, eksctl as shell scripts into the buildspec file

For the buildspec and other files check out the gdrive link

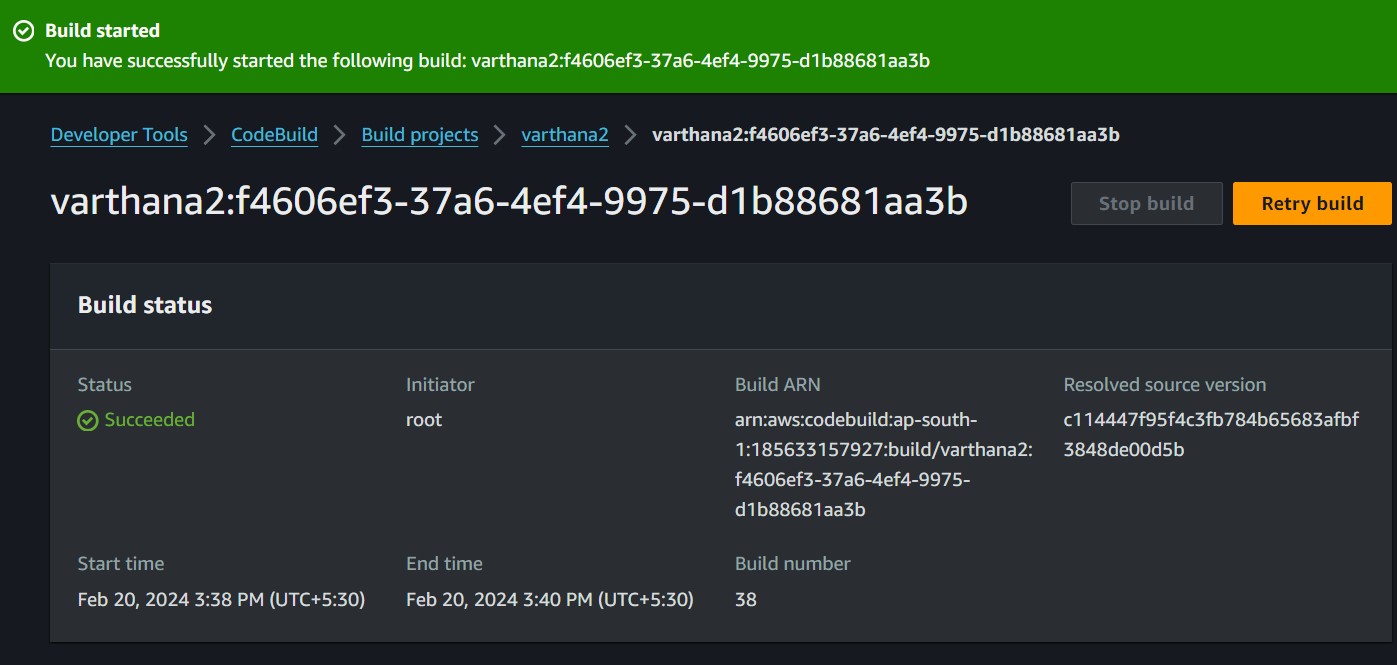
https://drive.google.com/drive/folders/1Yl1zPKiu31jEQCjqaXMG2eDH1o395u1s?usp=drive\_link

1. navigate to code build and choose to start build the app

The image is created



Pushed to ECR

And the build is successful

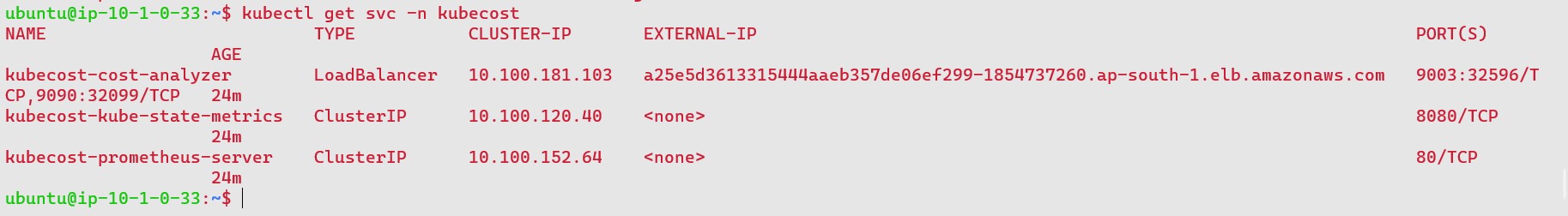
1. Navigate back to the terminal and access the pods and verify the nodes are running

Kubectl get nodes

1. Kubectl get pods -n kubecost



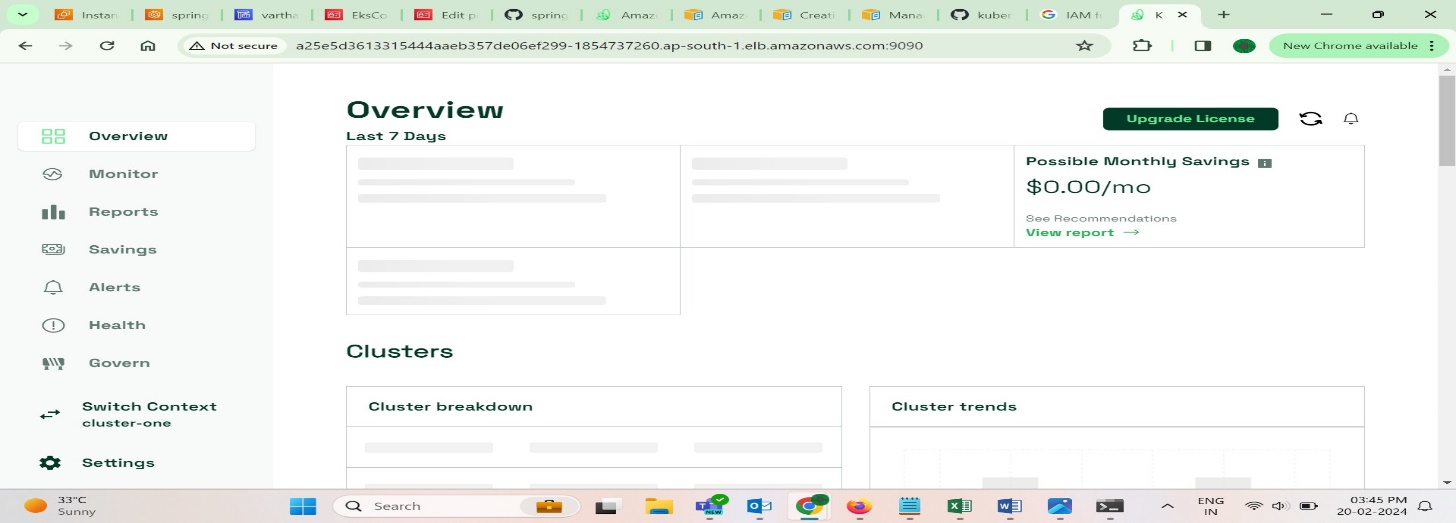
1. Kubectl get svc -n kubecost



1. Kubectl get pvc -n kubecost



1. Copy the loadbalancer dns and access it tinto the browser with port number :9090 to it



Done

Link for files

https://drive.google.com/drive/folders/1Yl1zPKiu31jEQCjqaXMG2eDH1o395u1s?usp=drive\_link