# DevOps Project

## Problem Statement:

Create an end-to-end CI/CD pipeline in AWS platform using Jenkins as the orchestration tool, GitHub as scm, maven as the build tool, deploy in a docker instance and create a docker image, store the docker image in ECR, Kubernetes deployment using ECR image. Build sample java web app using maven.

## Approach:

## Requirements:

- CI/CD declarative pipeline System
- Git - local version control system.
- GitHub - As Distributed version control system.
- Jenkins - Continuous Integration tool.
- Maven - As a Build Tool.
- docker -Containerization build tool using ECR
- Kubernetes - As Container Management Tool

## Step-1:

- Setup CI/CD with GitHub, Jenkins, Maven & Tomcat.
- Setup Jenkins
- Setup & Configure Maven, Git.
- Setup Tomcat Server.
- Integrating GitHub, Maven, Tomcat Server with Jenkins
- Create a CI and CD Job.
- Test the Deployment

**Step-2:**

- Setup CI/CD with GitHub, Jenkins, Maven & Docker.
- Setting up the docker Environment.
- Create an Image and Container on Docker Host.
- Integrate Docker Host with Jenkins.
- Create CI/CD Job on Jenkins to build and deploy on container.

**Step-3:**

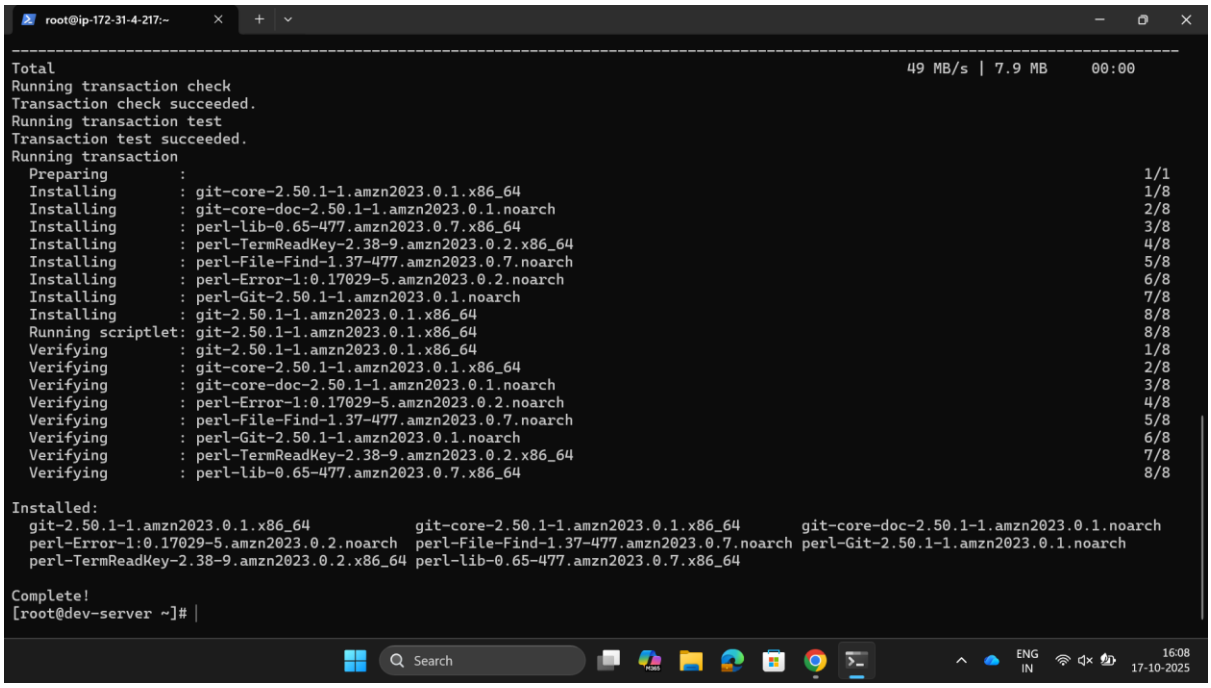
- Build and Deploy on Container.
- CI/CD with GitHub, Jenkins, Maven & Kubernetes.
- Setting up the Kubernetes (EKS).
- Write pod service and deployment manifest file.
- CI/CD Job to build code on Jenkins & Deploy it on Kubernetes.

**Step-4:**

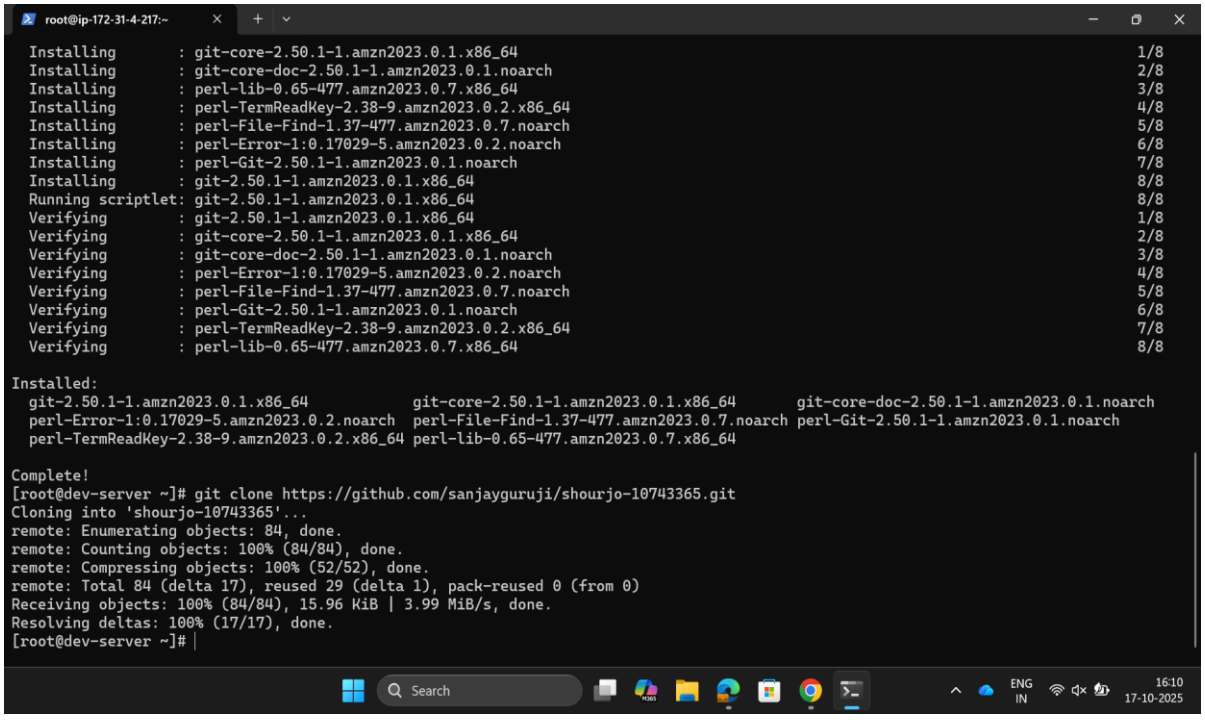
- Deploy artifacts on the Kubernetes
- Write codes in the artifacts of docker and Kubernetes which we want to run.
- Now build the code in Jenkins.
- Check in Kubernetes the pods are getting created or not.
- Now copy the service IP and paste it in the browser and check the output.

## Solution:

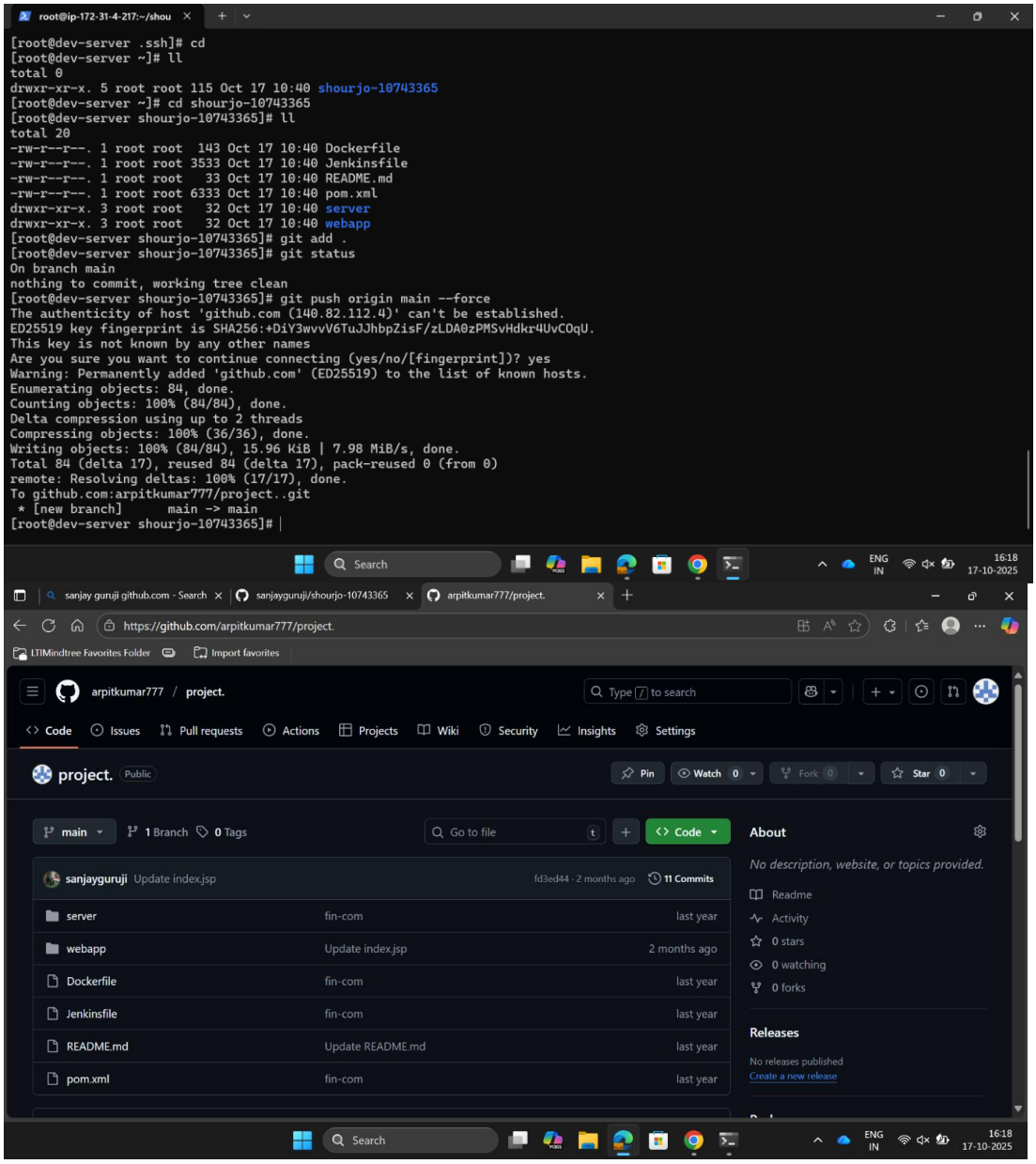
Creating an EC2 instance for Git operation.



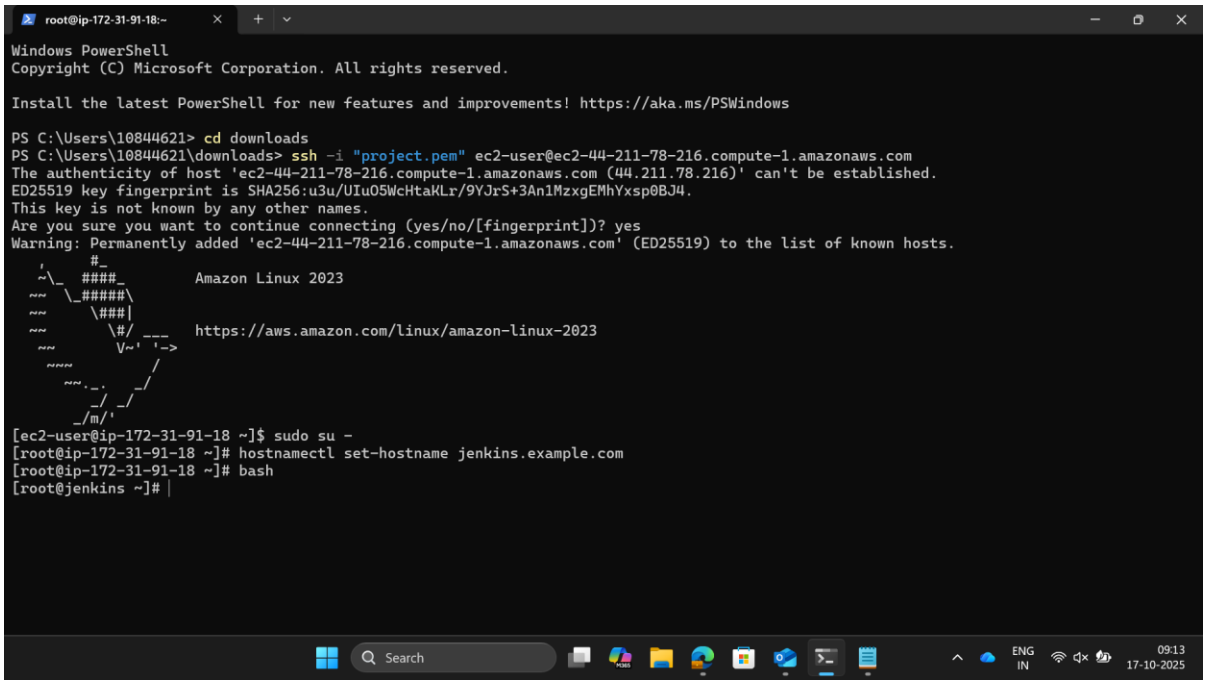

Code files are being cloned.



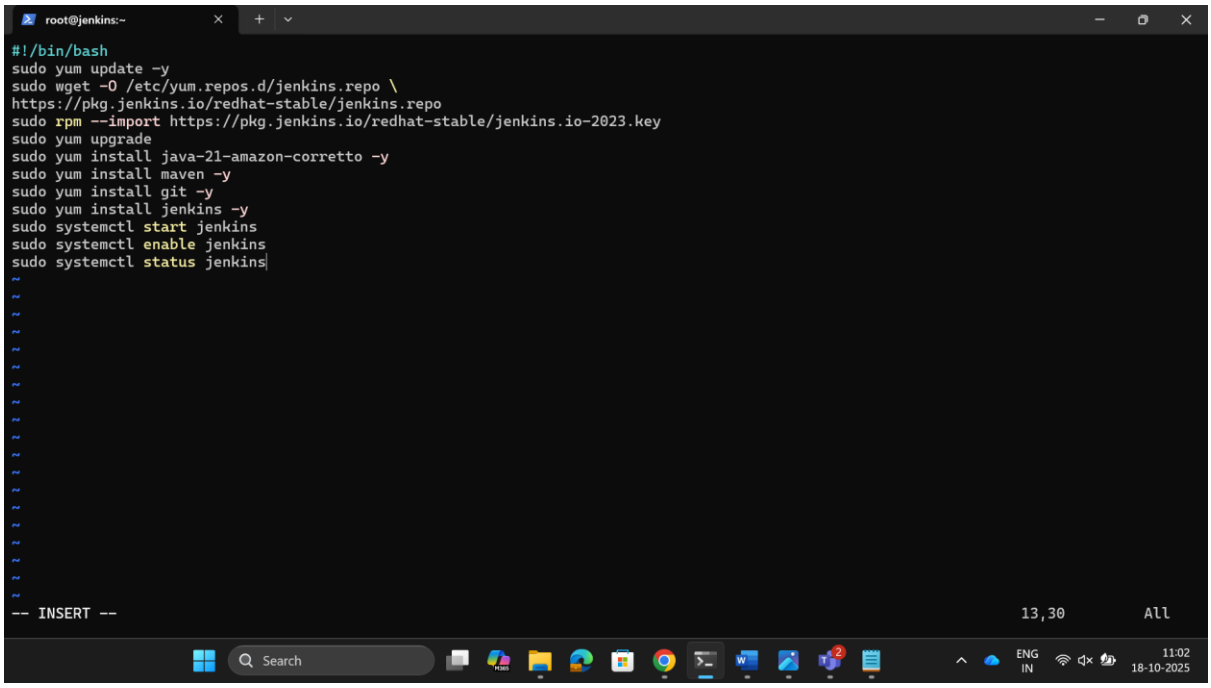
We can see that the files are being cloned.



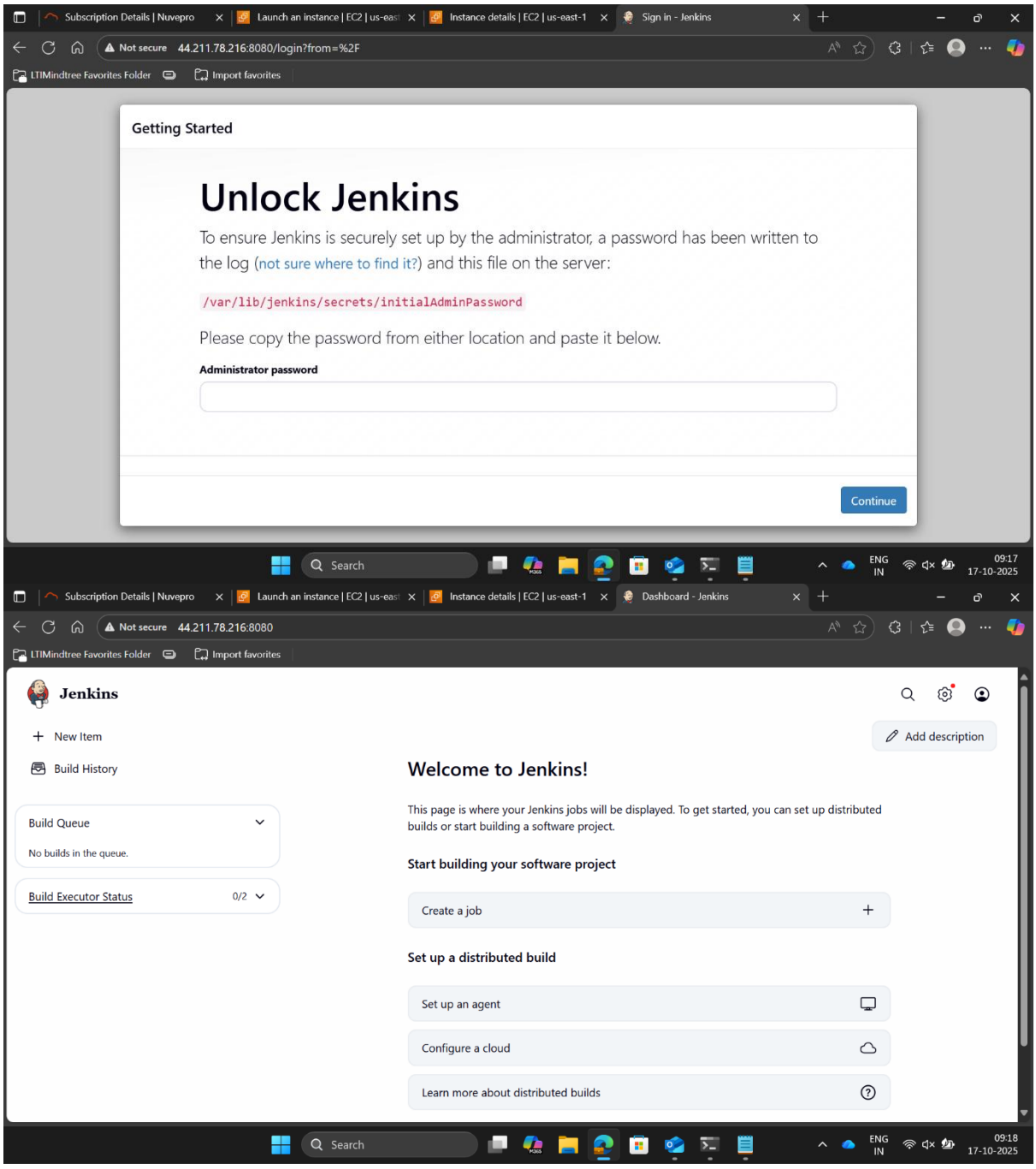
Creating a new  EC2 instance for Jenkins setup.



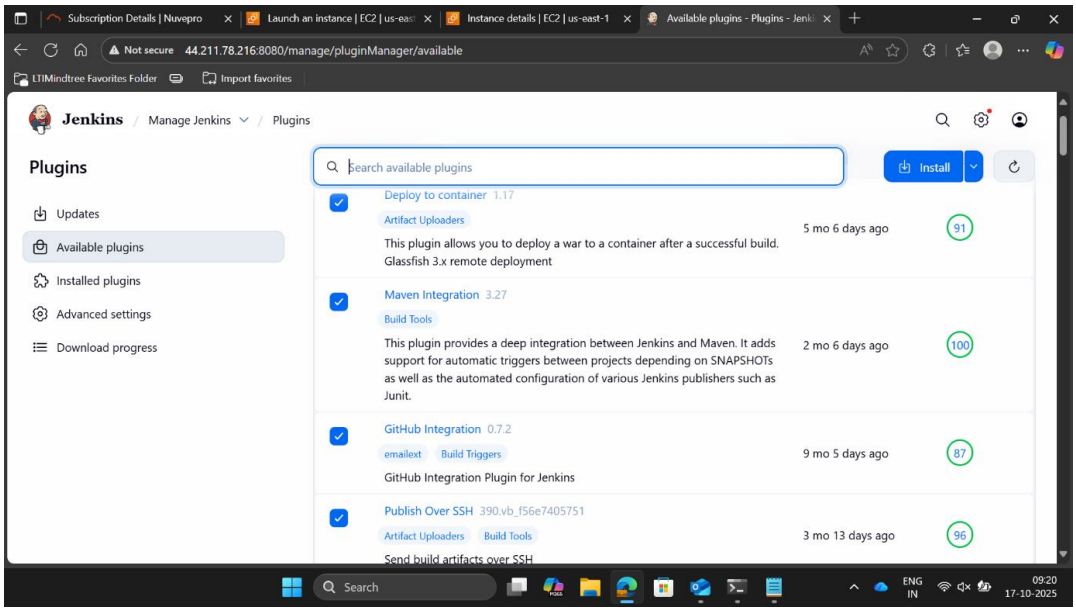

Jenkins script file for Jenkins setup.



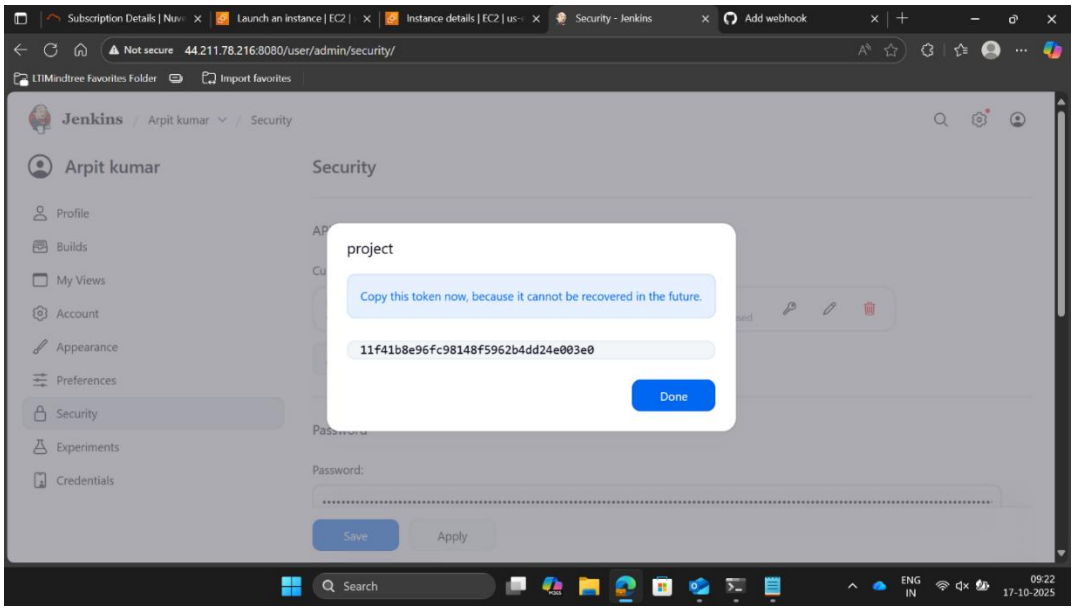
Jenkins setup is being done.



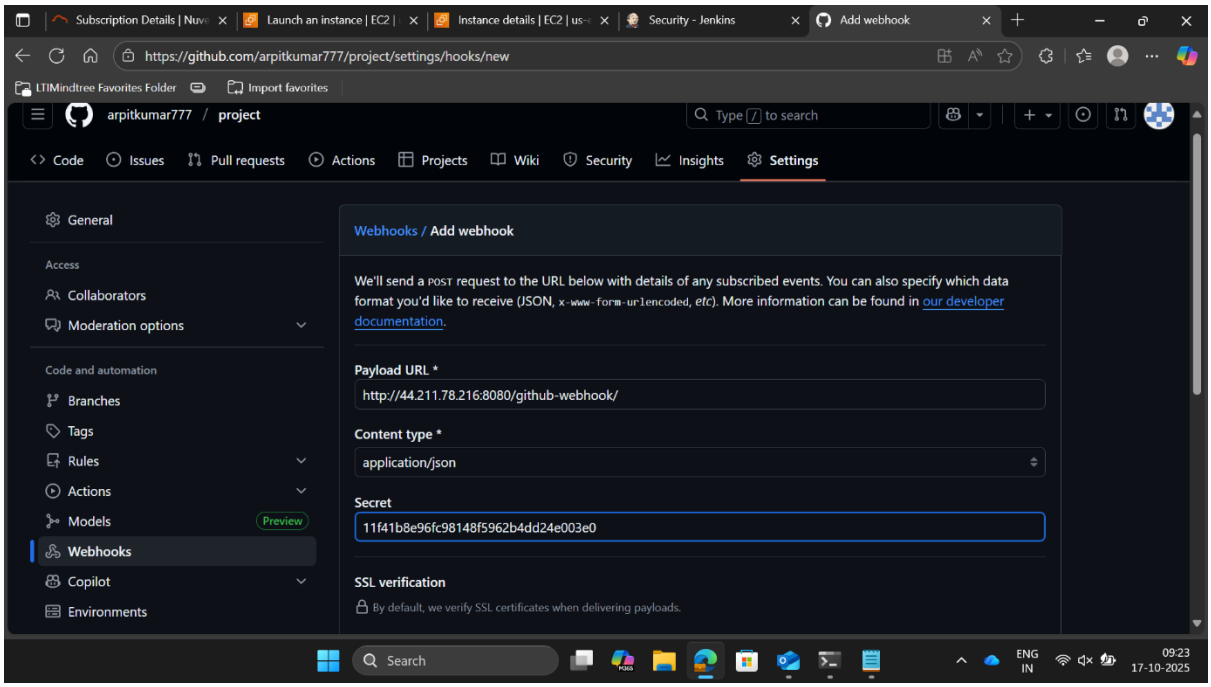
Installing the required plugins in Jenkins.



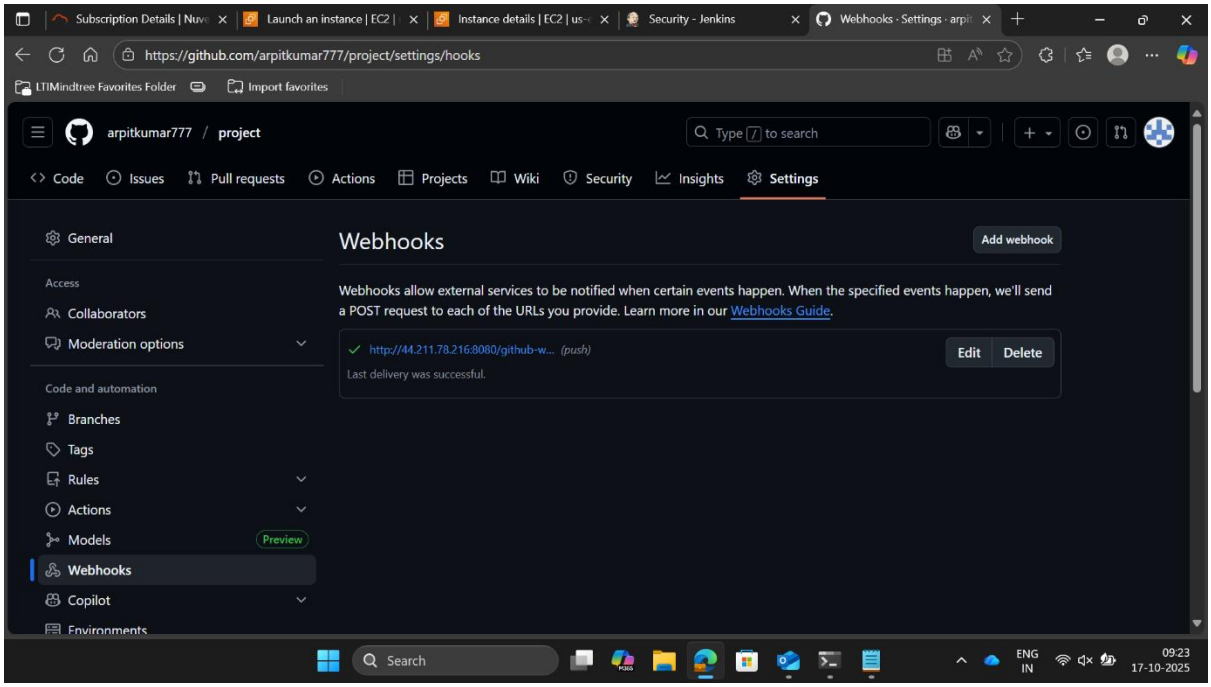

Creating security key in Jenkins



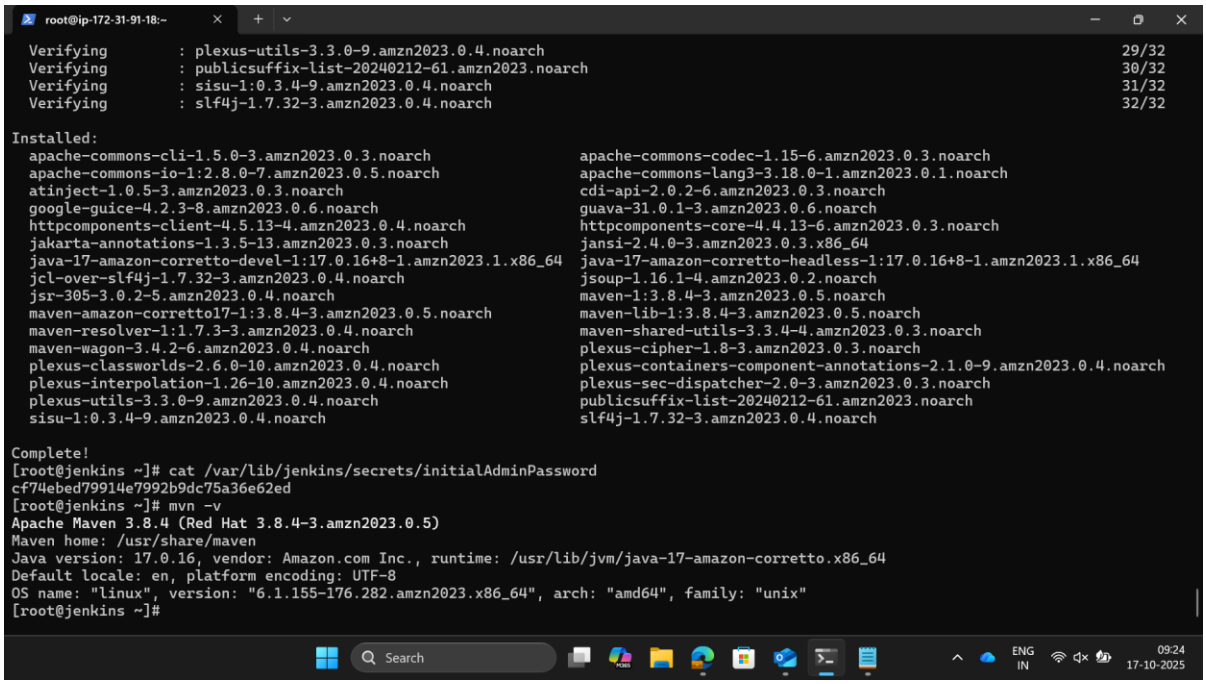
Adding webhook in GitHub.



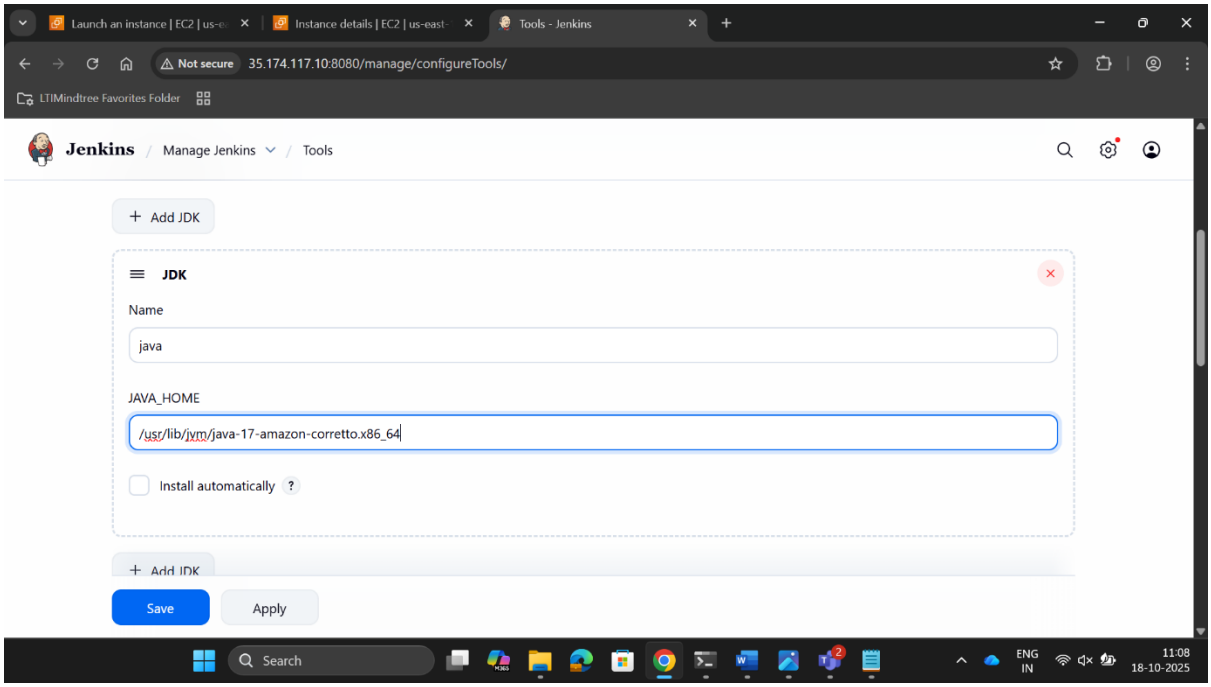

Webhook is successfully being added.



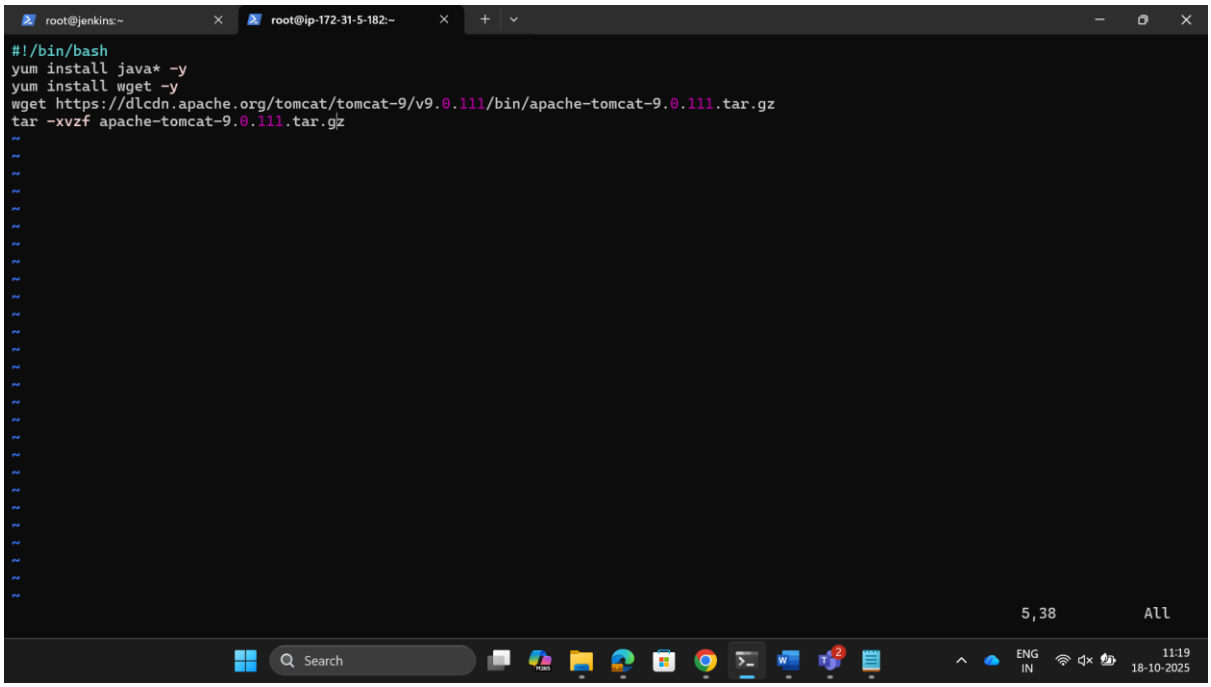
Getting java and maven path in java Jenkins terminal.



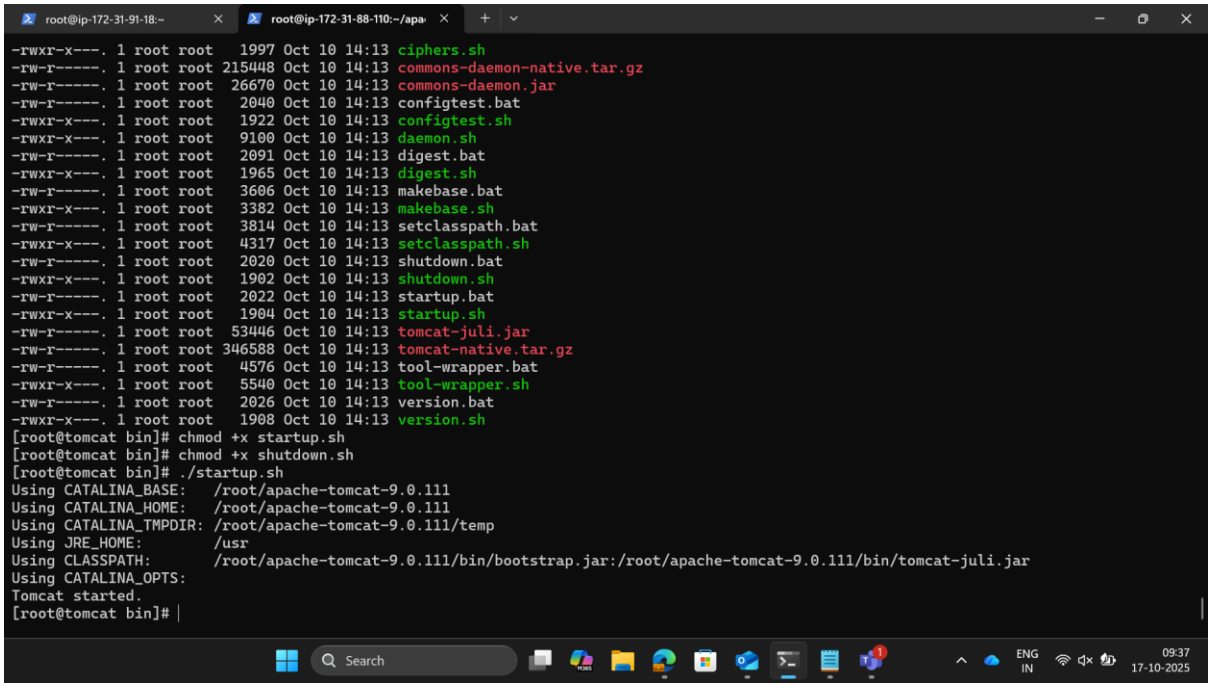

Adding java and maven path to Jenkins tool.



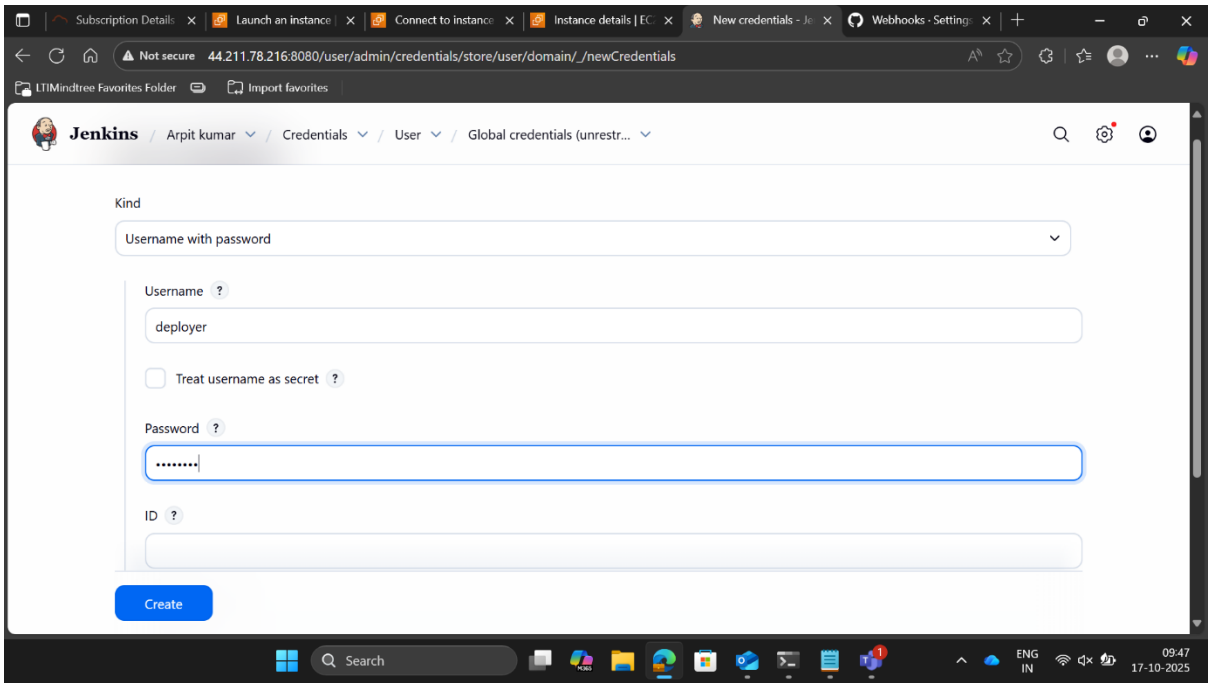
Creating new EC2 instance and setting up the tomcat server.



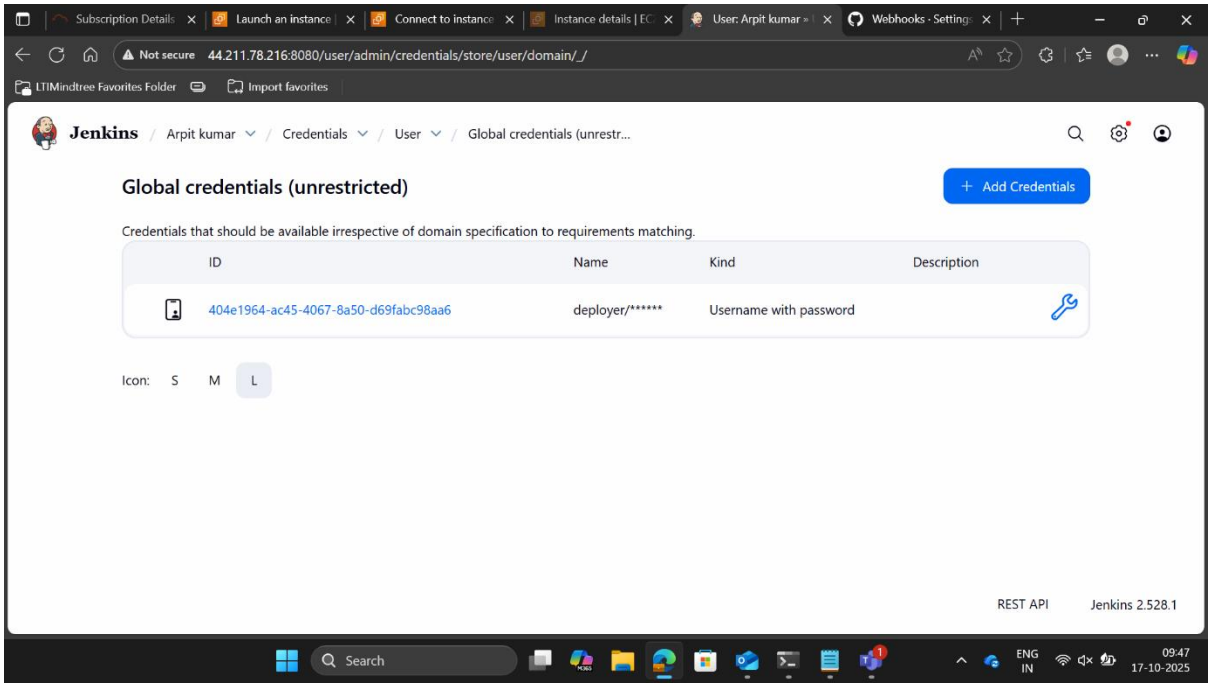

Giving permissions for starting up and shutdown of tomcat server.



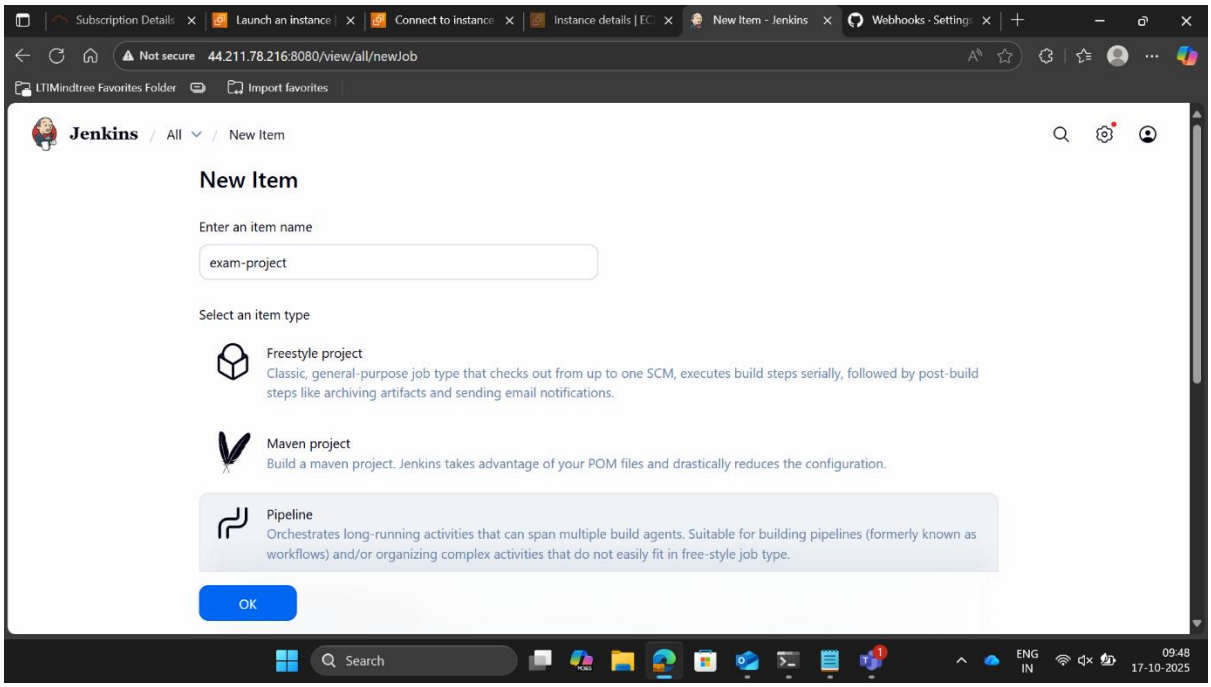
Setting up the global credentials in Jenkins.



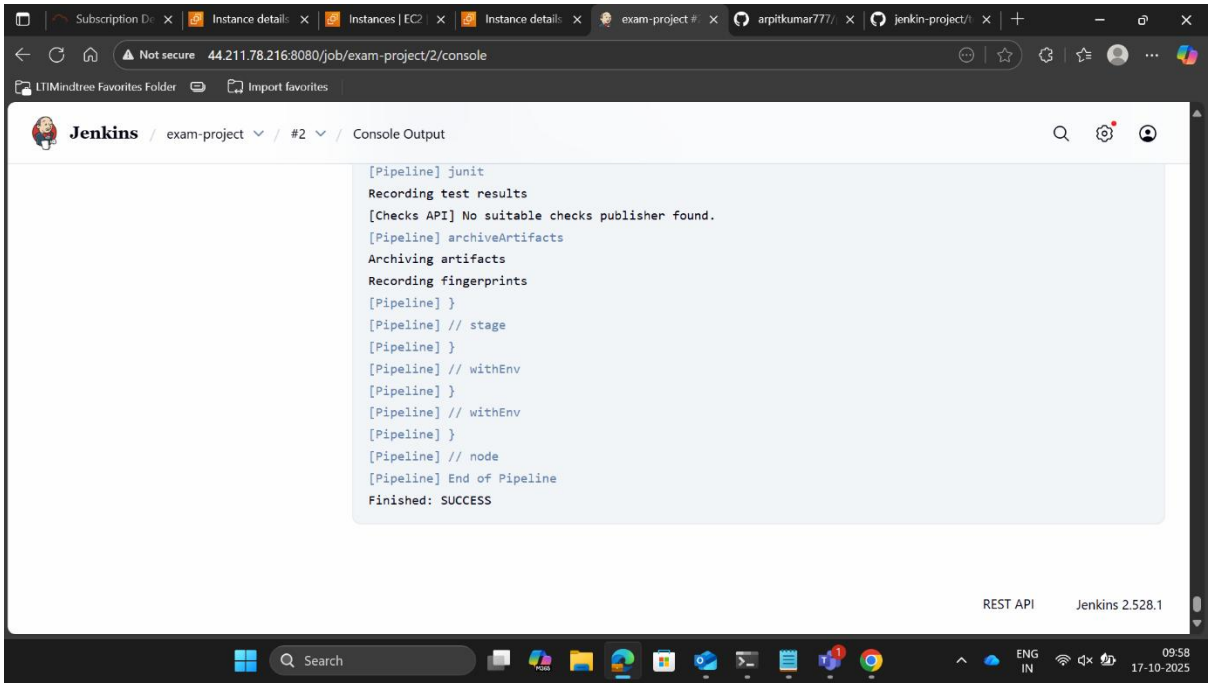

Credentials are being set up in Jenkins.



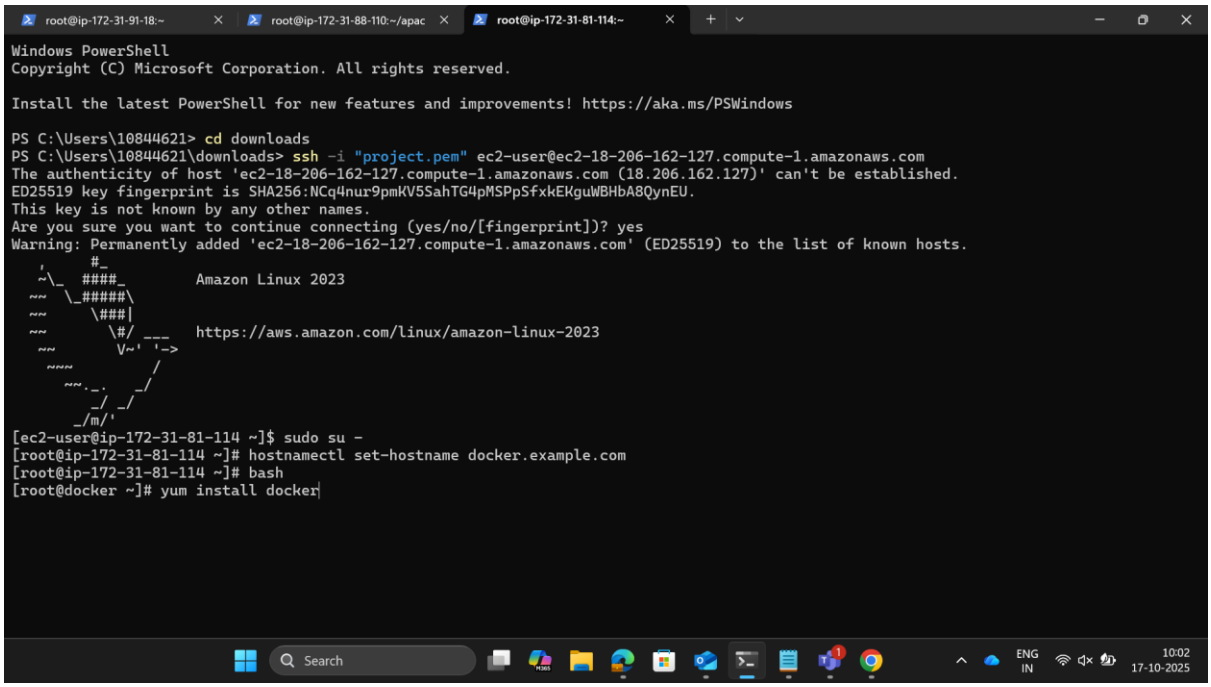
Now we are creating a pipeline project.



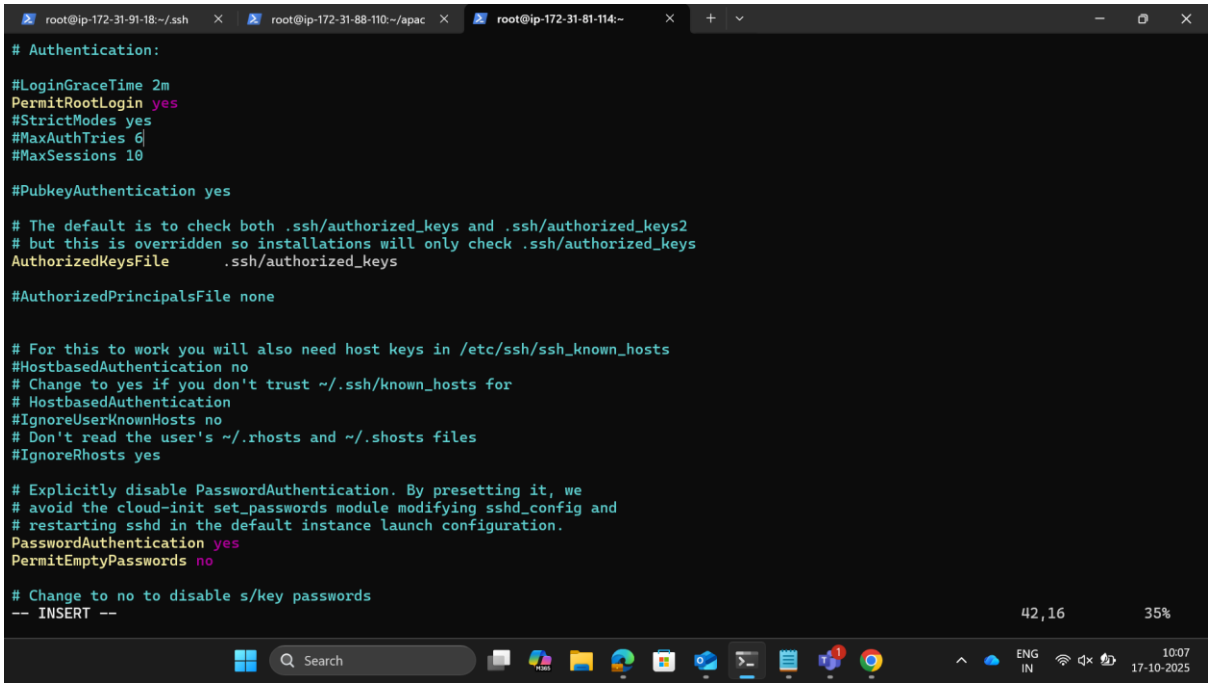

We can see that our build is being successfully finished till here.



Creating EC2 instance for docker installation.

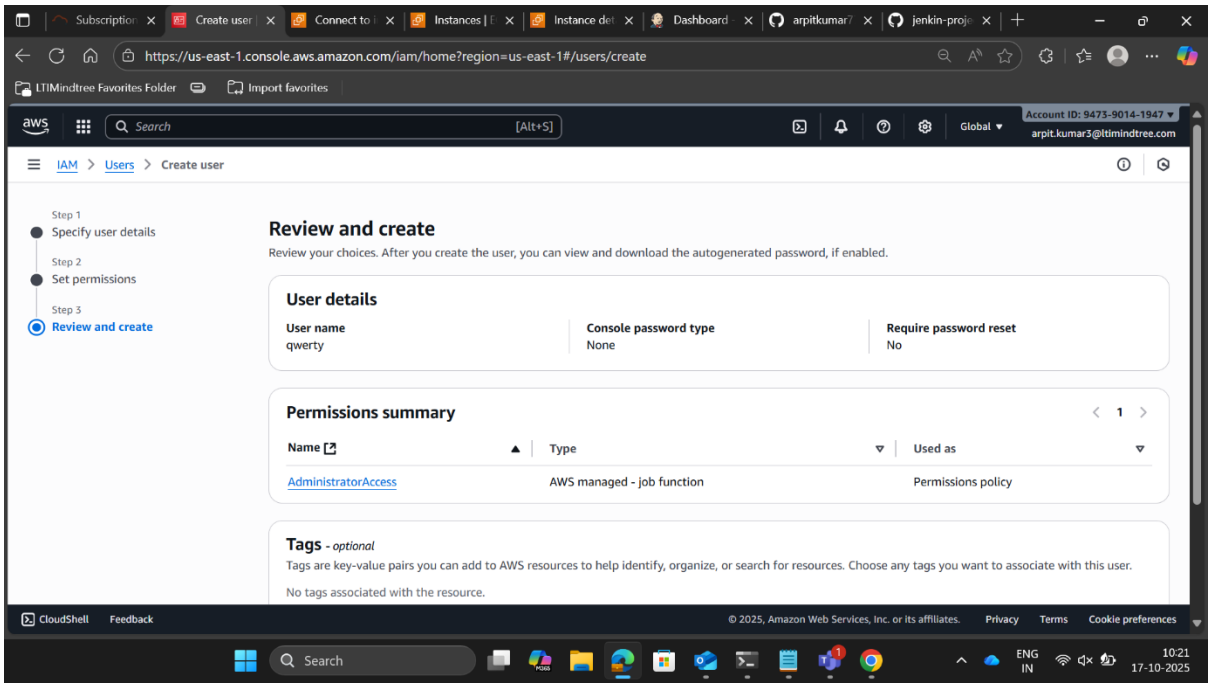



Giving password authentication permissions.

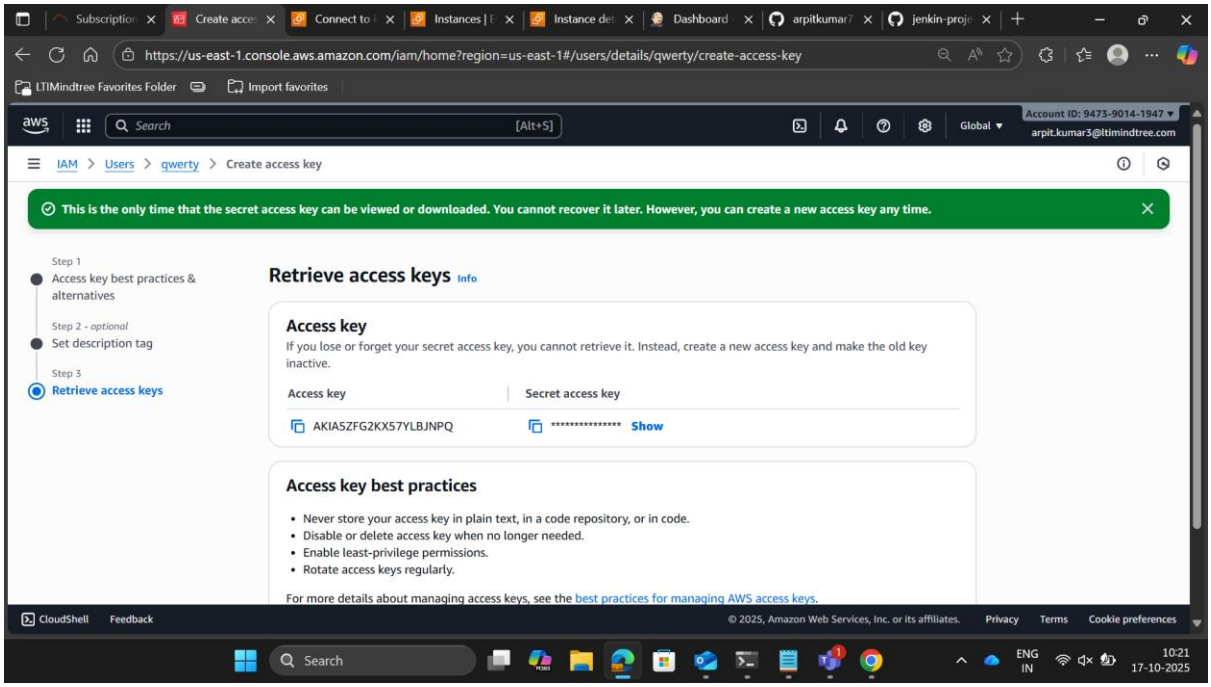


Creating the user and giving administrator full access permission

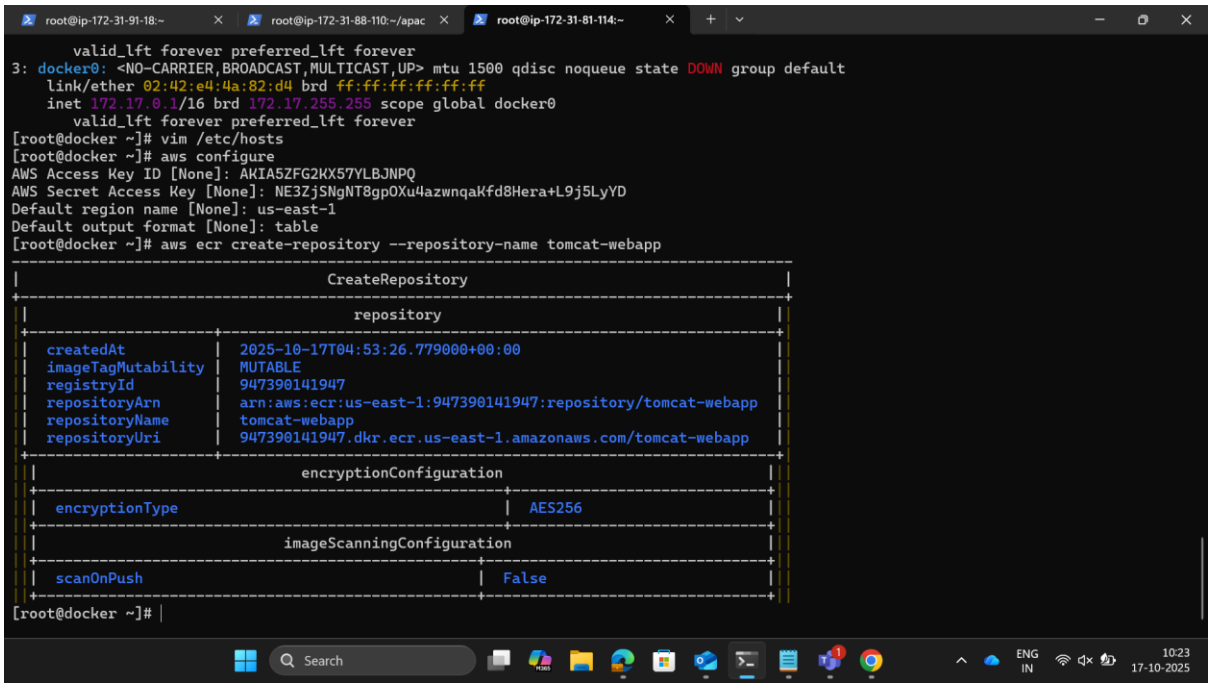
.



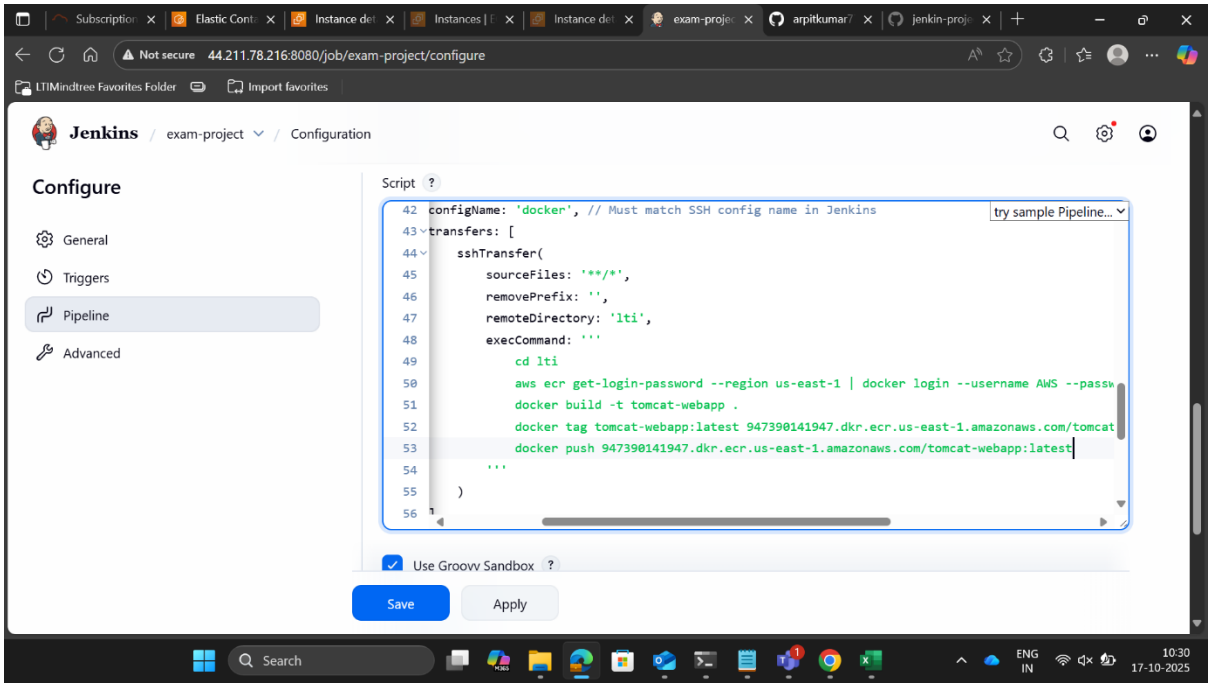

Generating access key for aws configuration.



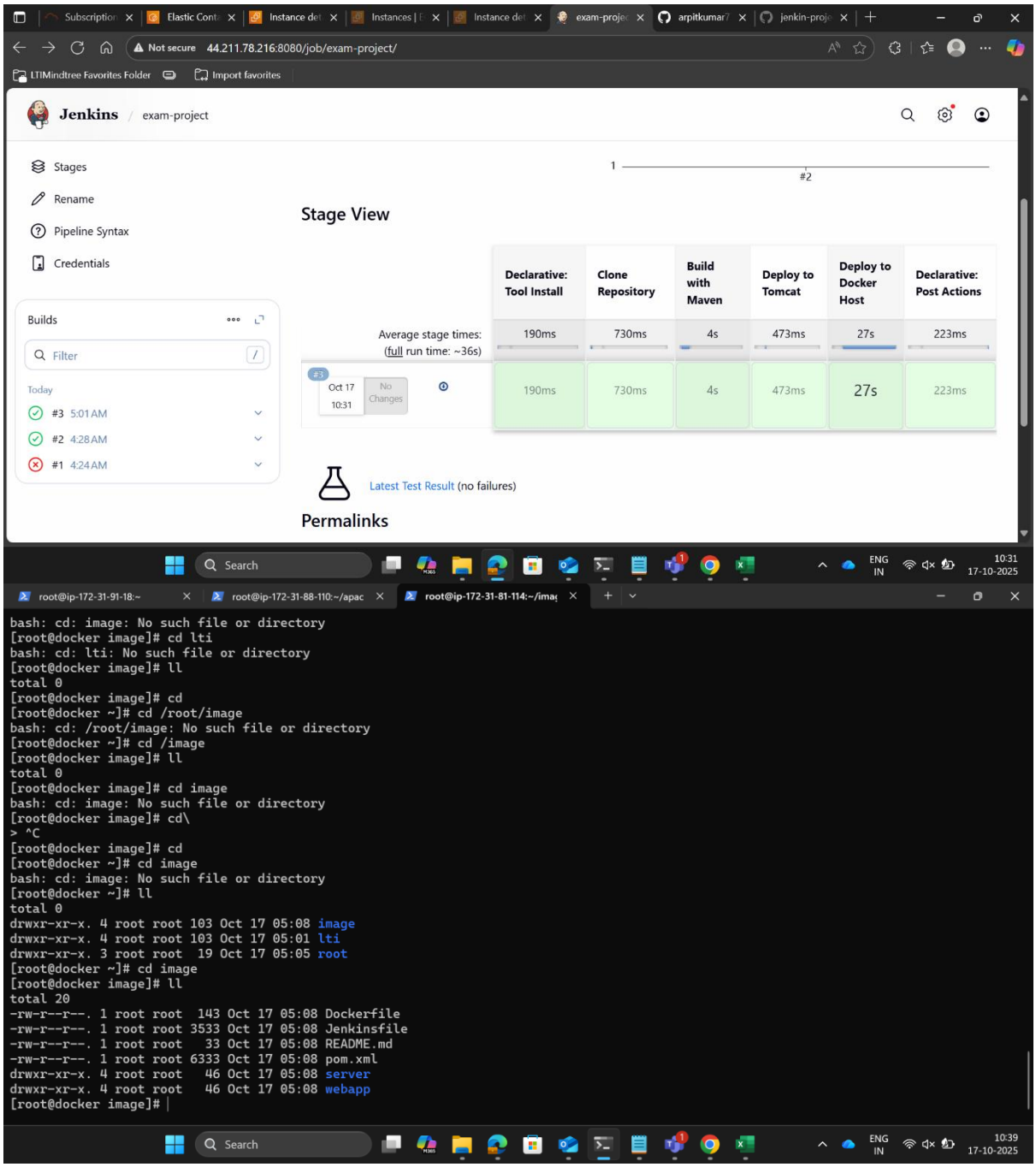
## Creating AWS ECR



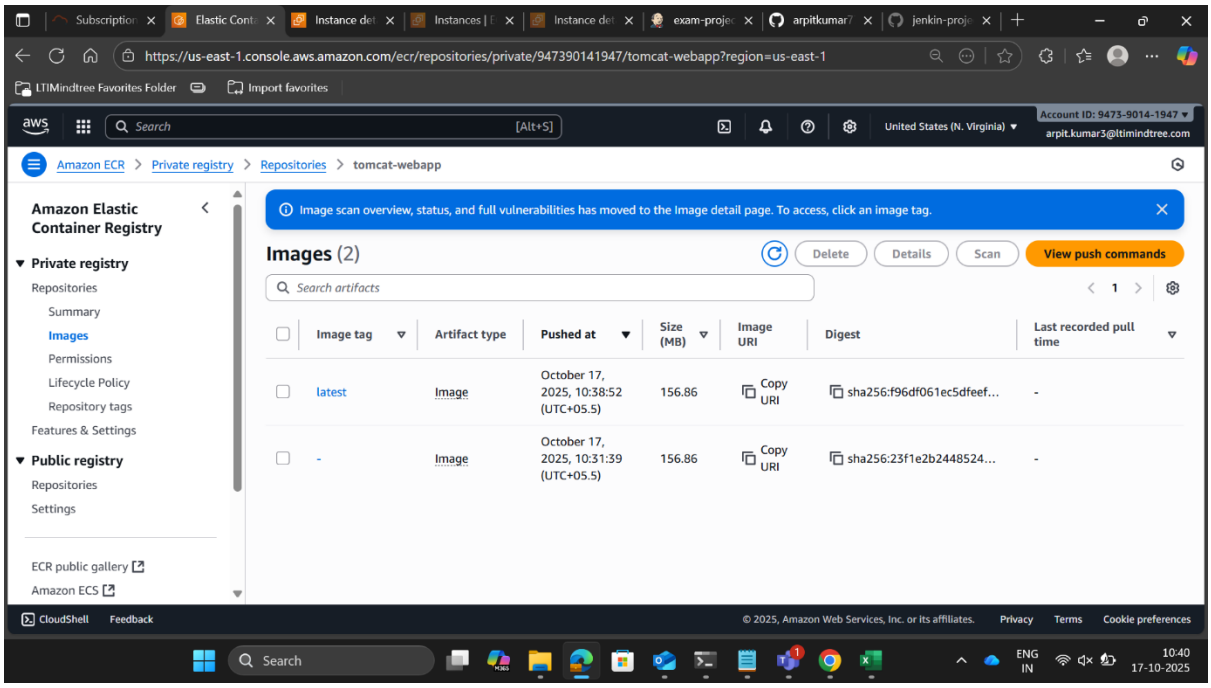

## Creating image inside docker.



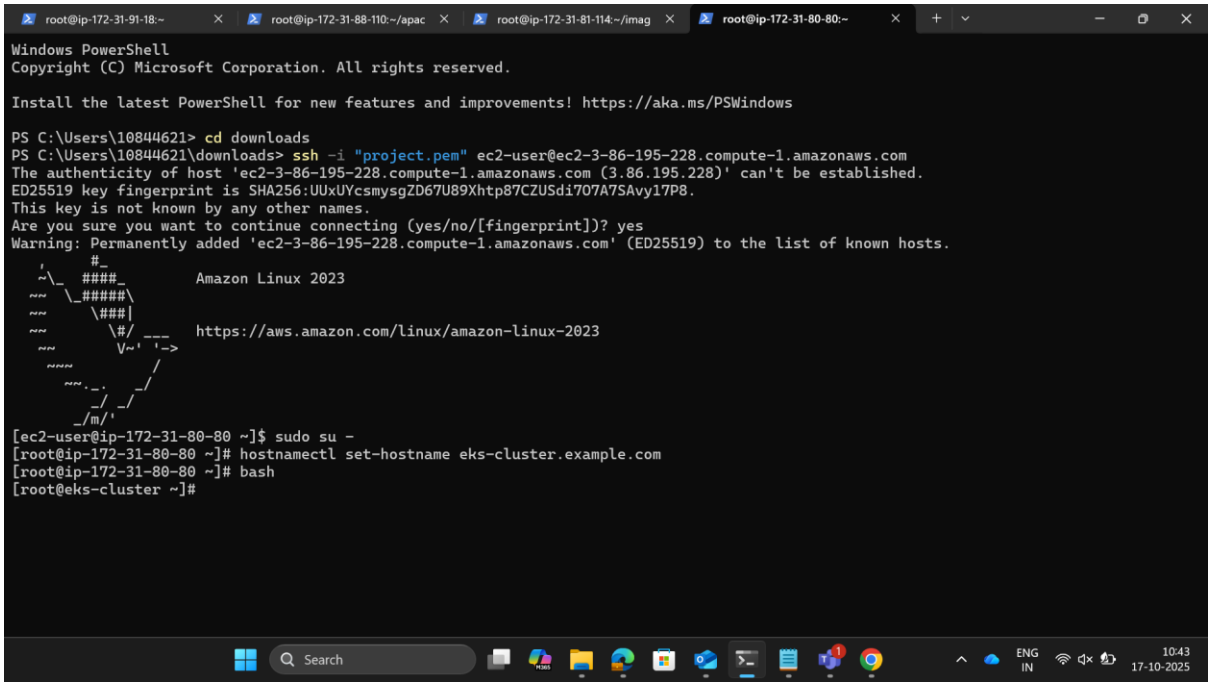
Our project is successful build for docker image .



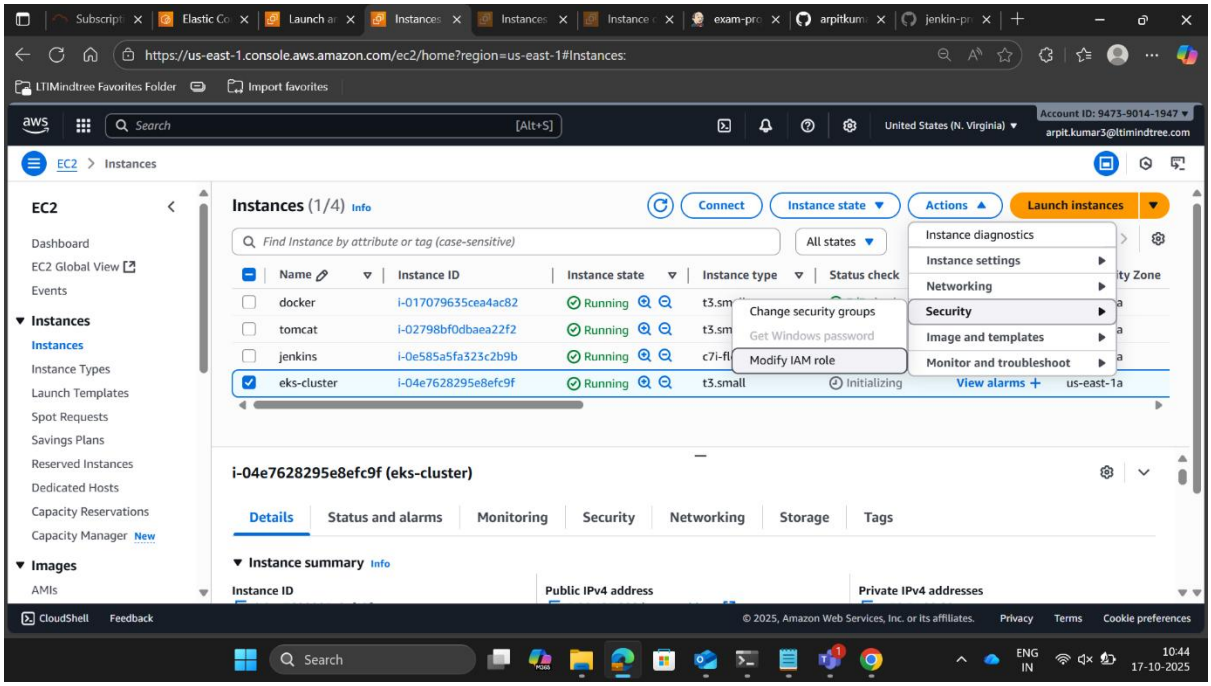
We can see the image in the amazon ECR.



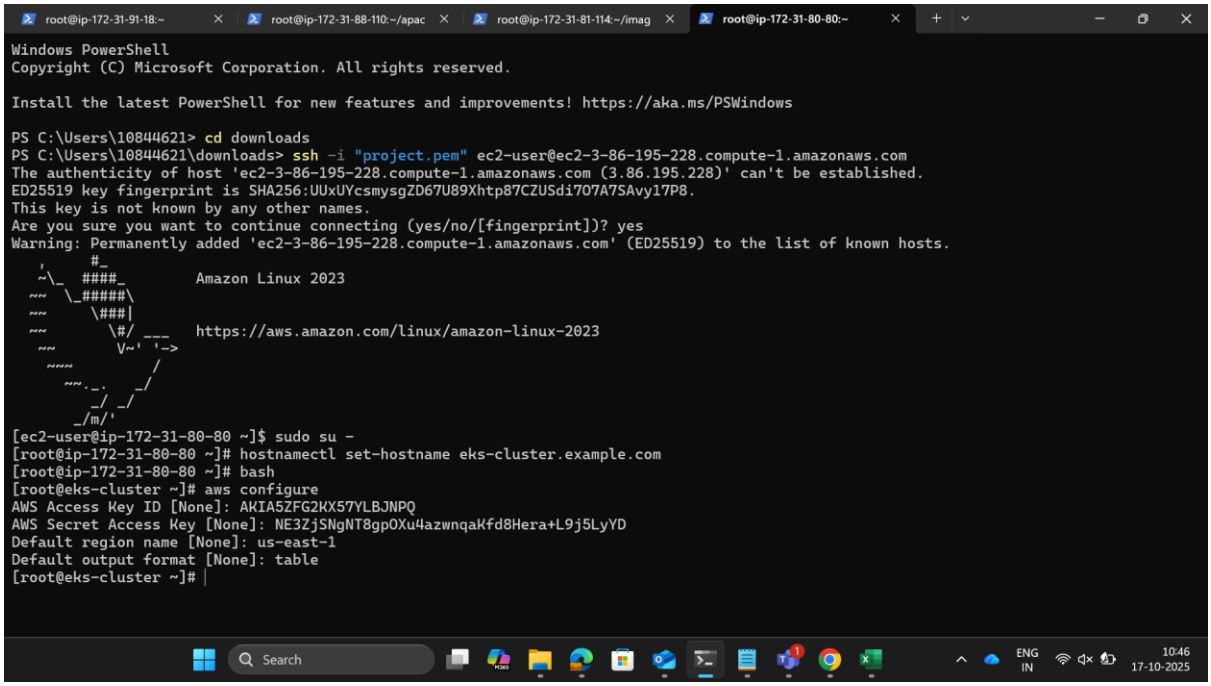

Creating EC2 instance for eks cluster



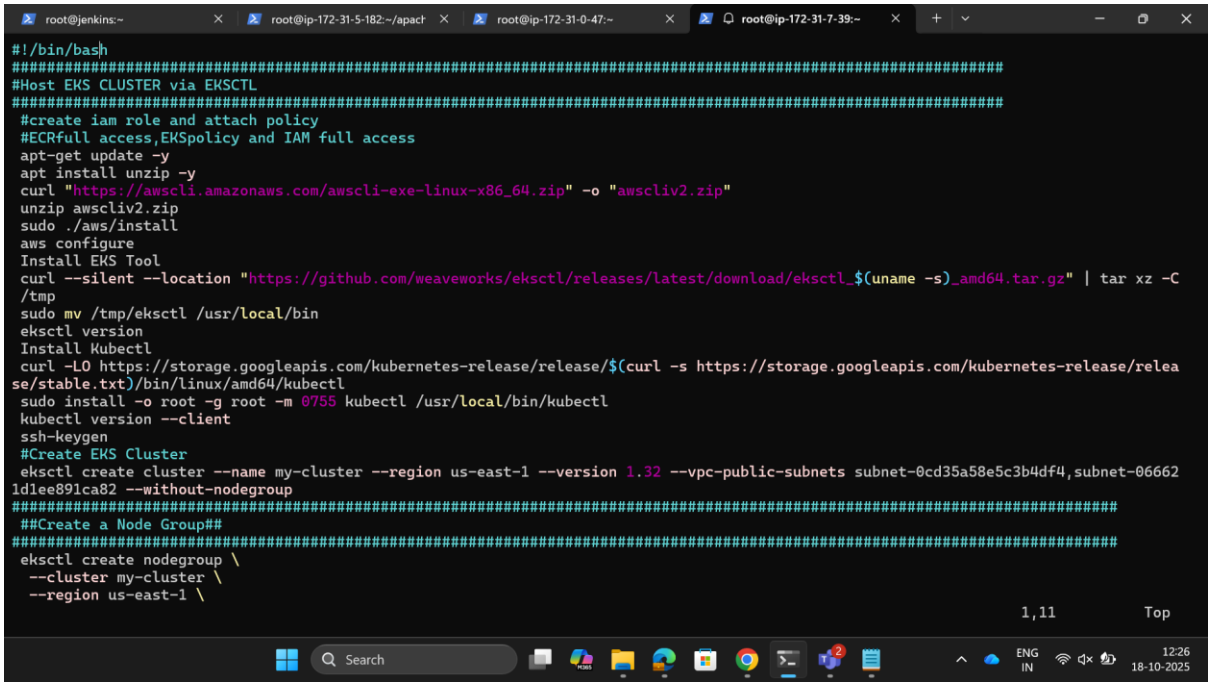
Assigning role to the instance.




AWS configuration of EKS instance.

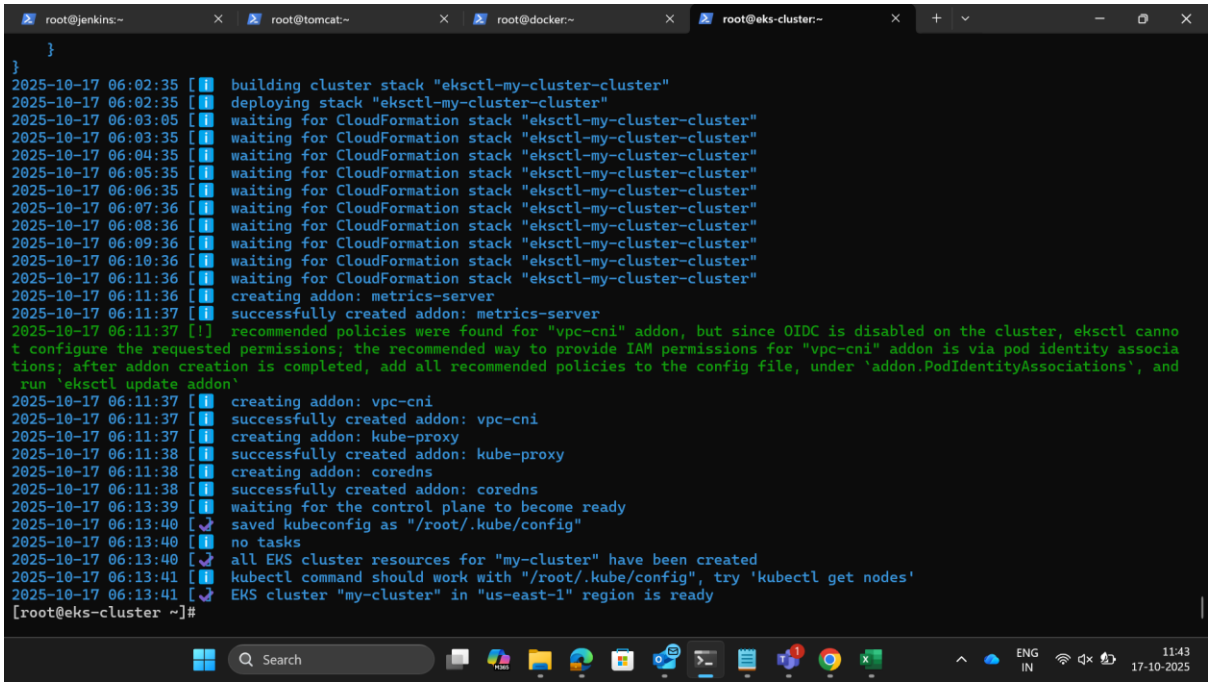


Script for setting up the EKS cluster EC2 instance.



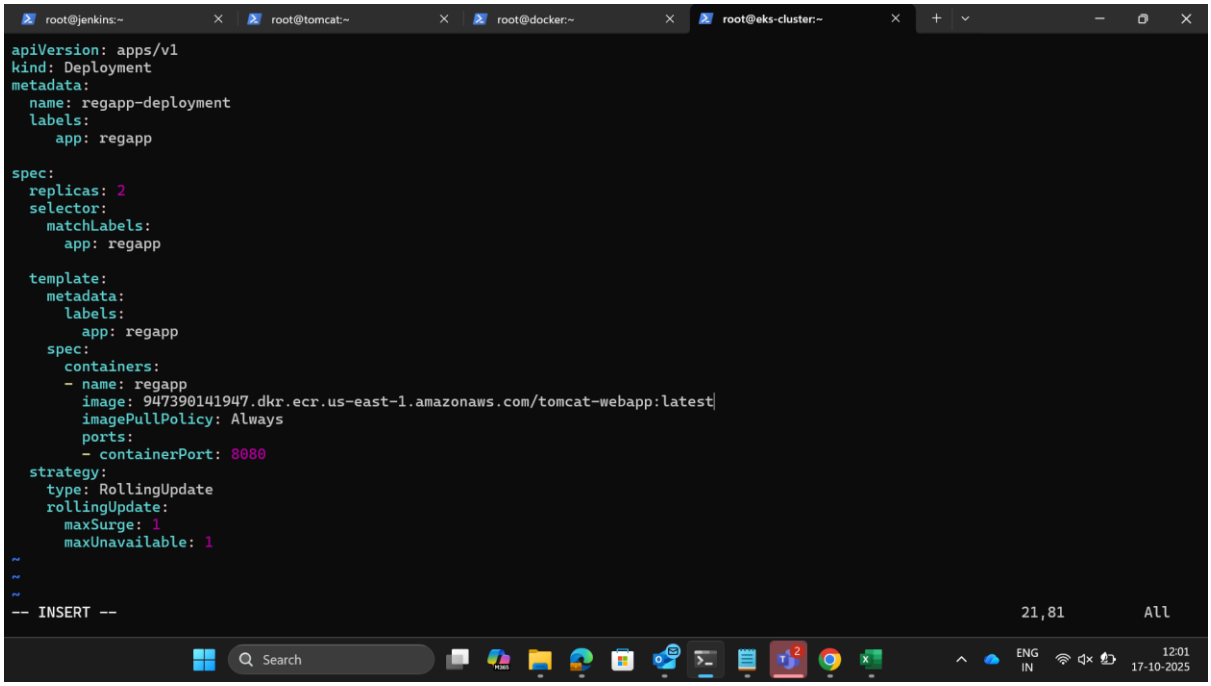
```
#!/bin/bash
##############################################################################################################################
#Host EKS CLUSTER via EKSCTL
##############################################################################################################################
 #create iam role and attach policy
 #ECRfull access,EKSpolicy and IAM full access
 apt-get update -y
 apt install unzip -y
 curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
 unzip awscliv2.zip
 sudo ./aws/install
 aws configure
 Install EKS Tool
 curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C
 /tmp
 sudo mv /tmp/eksctl /usr/local/bin
 eksctl version
 Install Kubectl
 curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/relea
se/stable.txt)/bin/linux/amd64/kubectl
 sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
 kubectl version --client
 ssh-keygen
 #Create EKS Cluster
 eksctl create cluster --name my-cluster --region us-east-1 --version 1.32 --vpc-public-subnets subnet-0cd35a58e5c3b4df4,subnet-06662
1d1ee891ca82 --without-nodegroup
##############################################################################################################################
 ##Create a Node Group##
##############################################################################################################################
 eksctl create nodegroup \
  --cluster my-cluster \
  --region us-east-1 \
                                                                                                              1,11          Top
```

Cluster is being created.

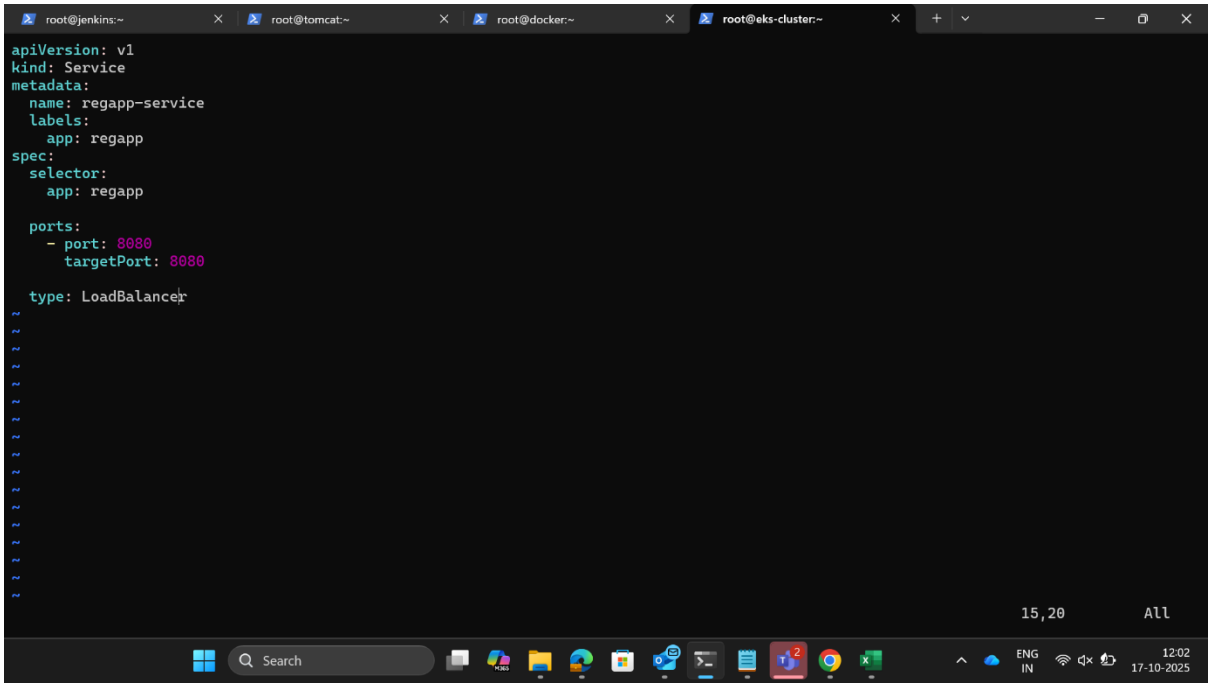


```
    }
}
2025-10-17 06:02:35 [ℹ]  building cluster stack "eksctl-my-cluster-cluster"
2025-10-17 06:02:35 [ℹ]  deploying stack "eksctl-my-cluster-cluster"
2025-10-17 06:03:05 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:03:35 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:04:35 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:05:35 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:06:35 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:07:36 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:08:36 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:09:36 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:10:36 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:11:36 [ℹ]  waiting for CloudFormation stack "eksctl-my-cluster-cluster"
2025-10-17 06:11:36 [ℹ]  creating addon: metrics-server
2025-10-17 06:11:37 [ℹ]  successfully created addon: metrics-server
2025-10-17 06:11:37 [!]  recommended policies were found for "vpc-cni" addon, but since OIDC is disabled on the cluster, eksctl canno
t configure the requested permissions; the recommended way to provide IAM permissions for "vpc-cni" addon is via pod identity associa
tions; after addon creation is completed, add all recommended policies to the config file, under `addon.PodIdentityAssociations`, and
 run `eksctl update addon`
2025-10-17 06:11:37 [ℹ]  creating addon: vpc-cni
2025-10-17 06:11:37 [ℹ]  successfully created addon: vpc-cni
2025-10-17 06:11:37 [ℹ]  creating addon: kube-proxy
2025-10-17 06:11:38 [ℹ]  successfully created addon: kube-proxy
2025-10-17 06:11:38 [ℹ]  creating addon: coredns
2025-10-17 06:11:38 [ℹ]  successfully created addon: coredns
2025-10-17 06:13:39 [ℹ]  waiting for the control plane to become ready
2025-10-17 06:13:40 [✔]  saved kubeconfig as "/root/.kube/config"
2025-10-17 06:13:40 [ℹ]  no tasks
2025-10-17 06:13:40 [✔]  all EKS cluster resources for "my-cluster" have been created
2025-10-17 06:13:41 [ℹ]  kubectl command should work with "/root/.kube/config", try 'kubectl get nodes'
2025-10-17 06:13:41 [✔]  EKS cluster "my-cluster" in "us-east-1" region is ready
[root@eks-cluster ~]#
```

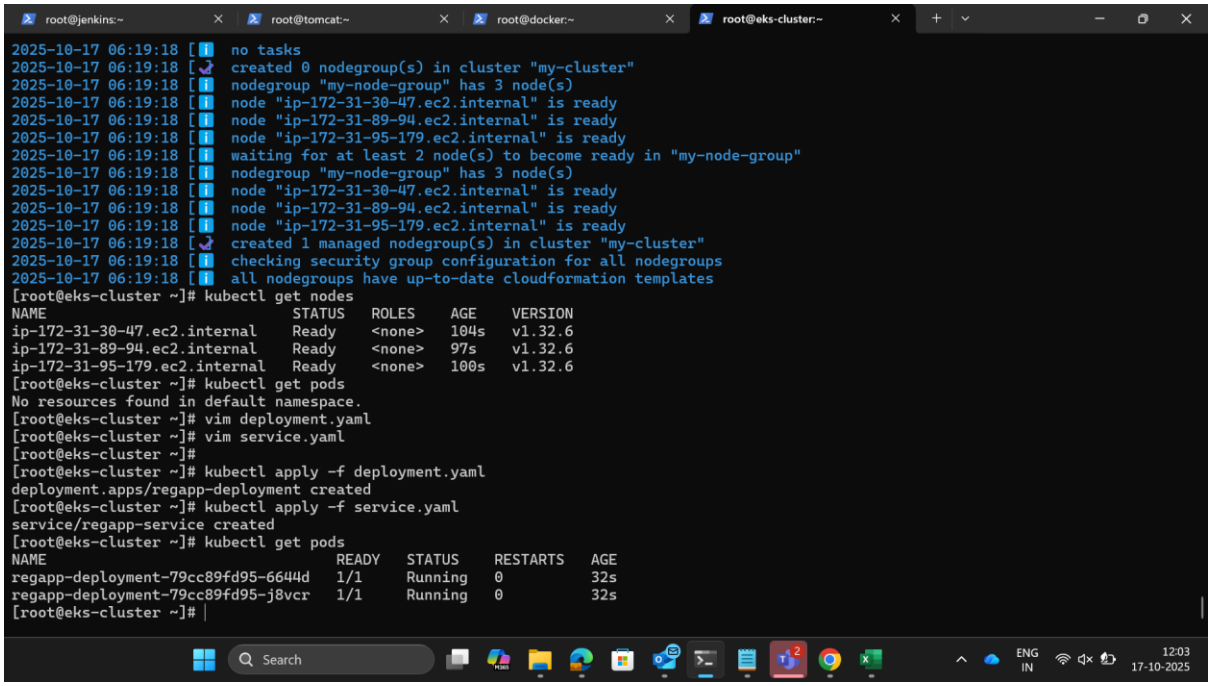
Creating deployment file for Autoscaling purpose.



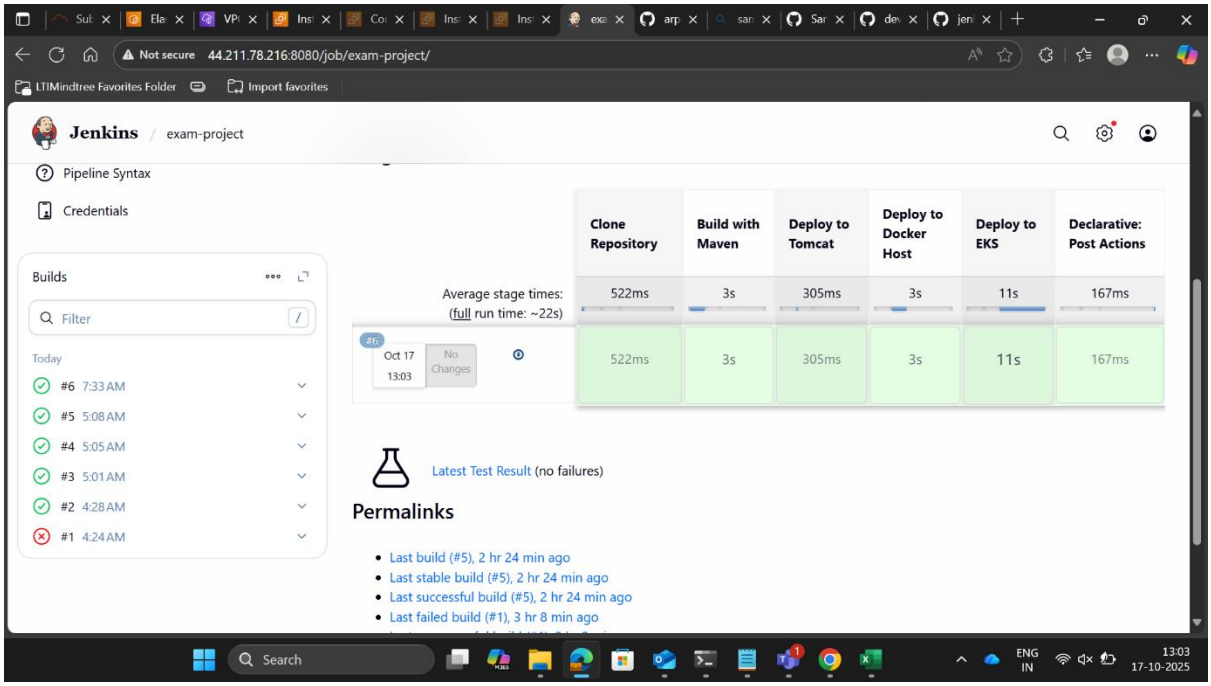

Creating service file for assigning i/p address for autoscaling group.



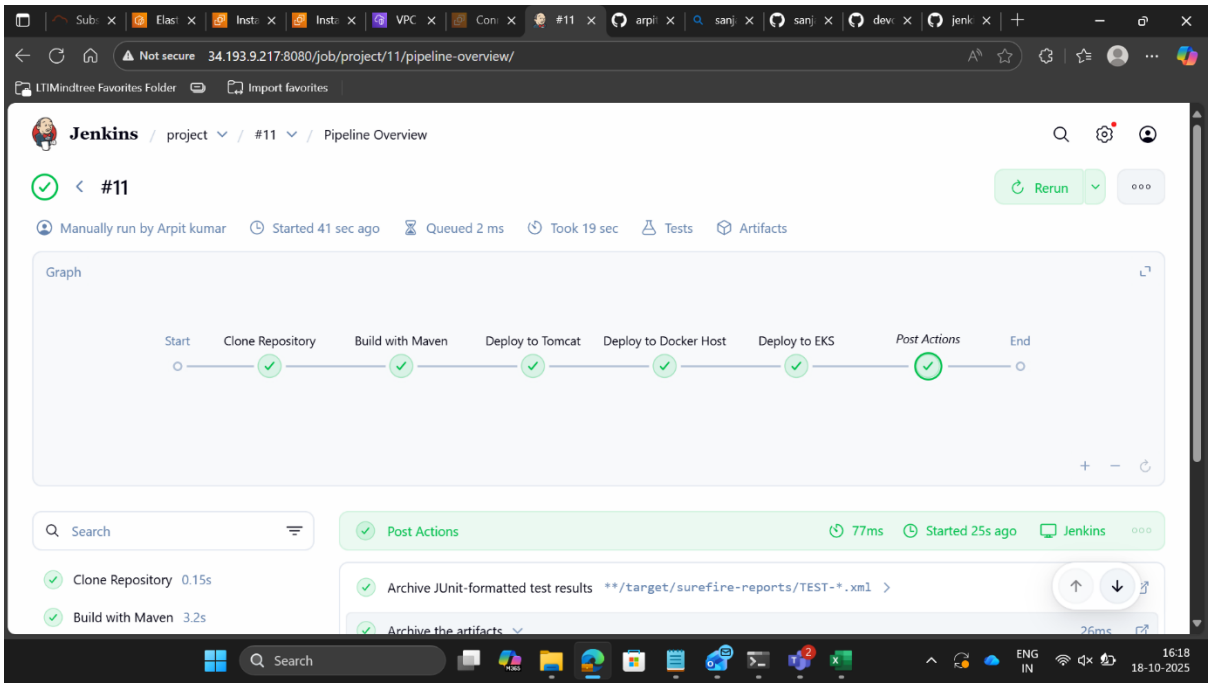
Running those two files deployment and service manifest files.



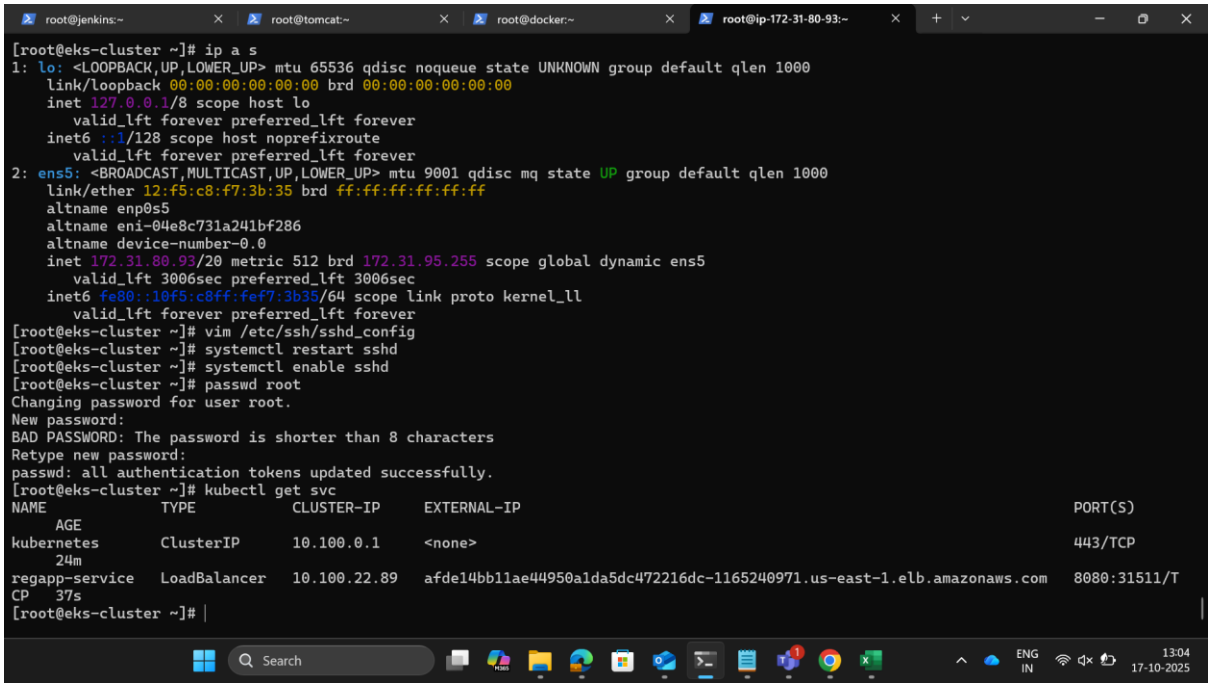

Our build is successful.



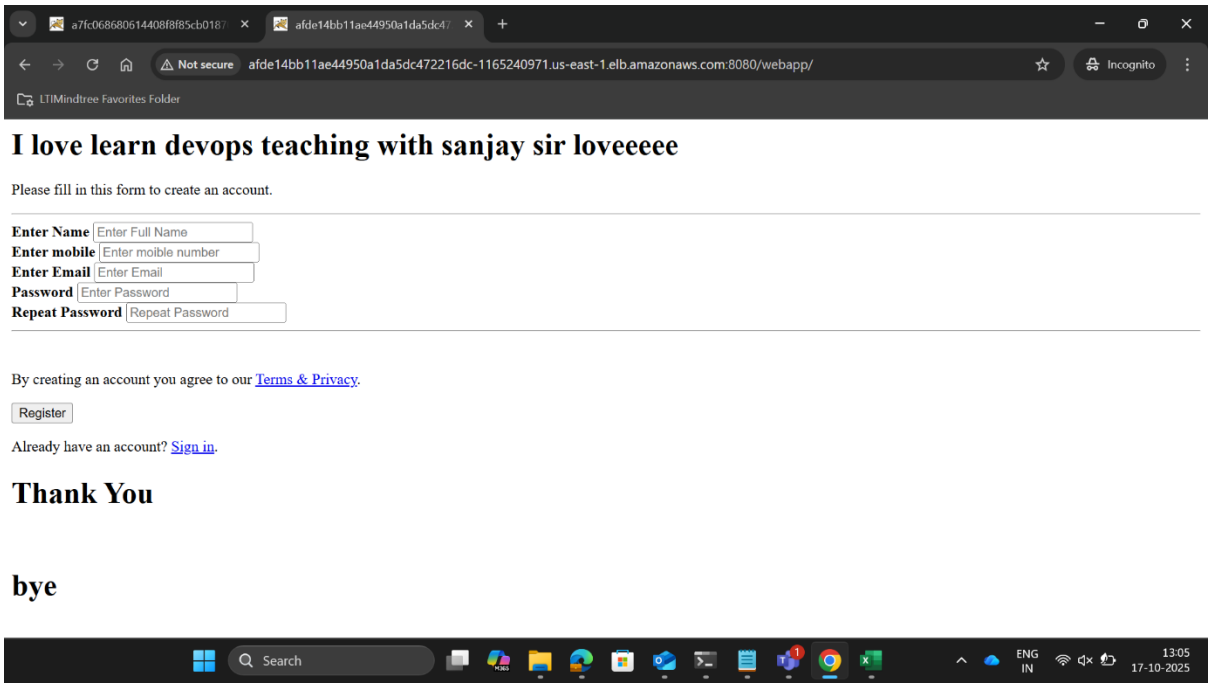
We can see the different stages of our build



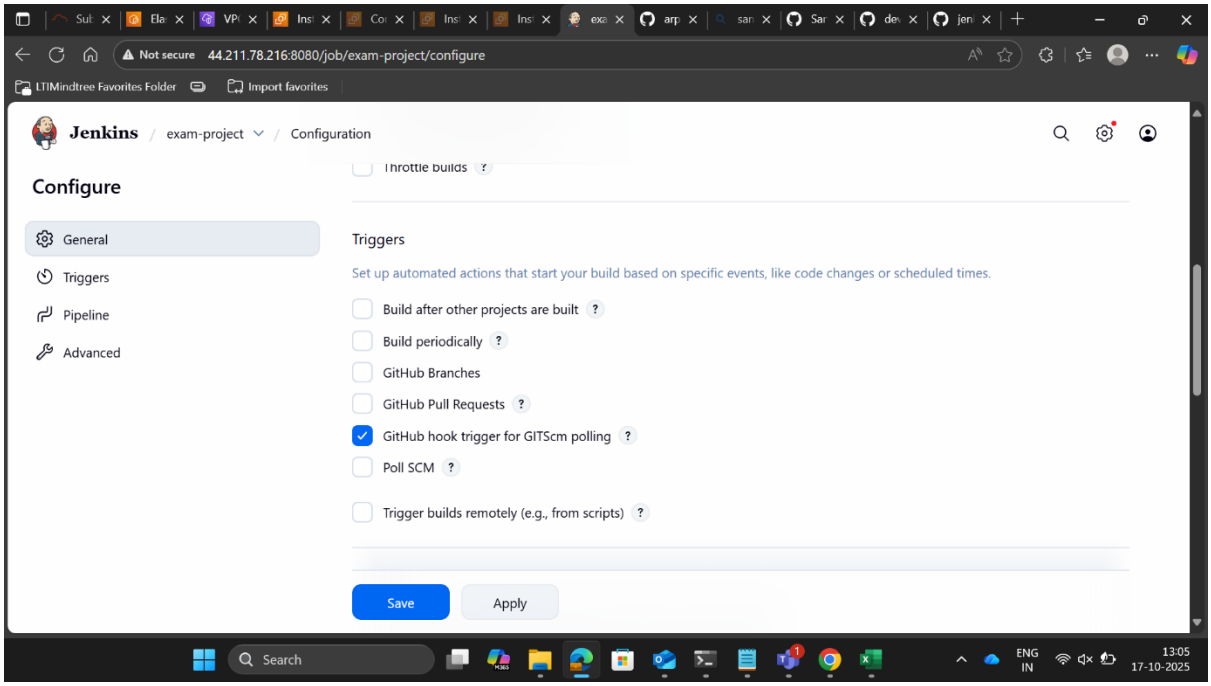

Nodes, Pods and service i/p is being shown.



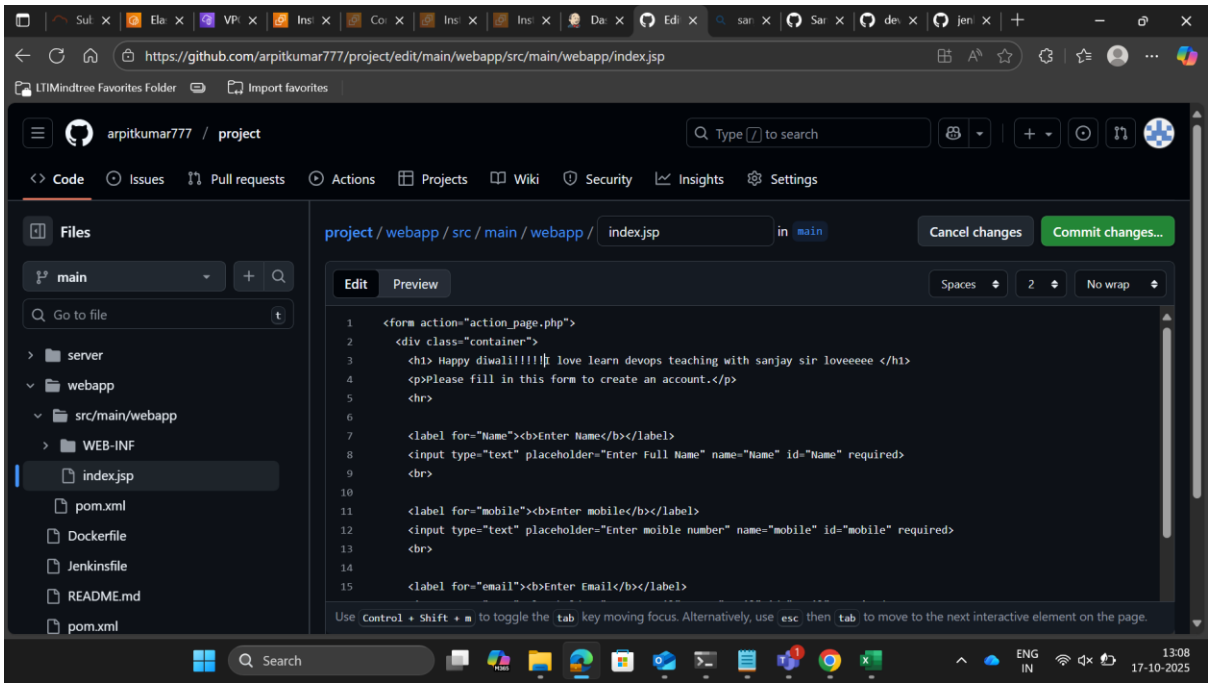
Final output is displayed by taking the service ip:8080/webapp/ which is being hosted on  the Load balancer.



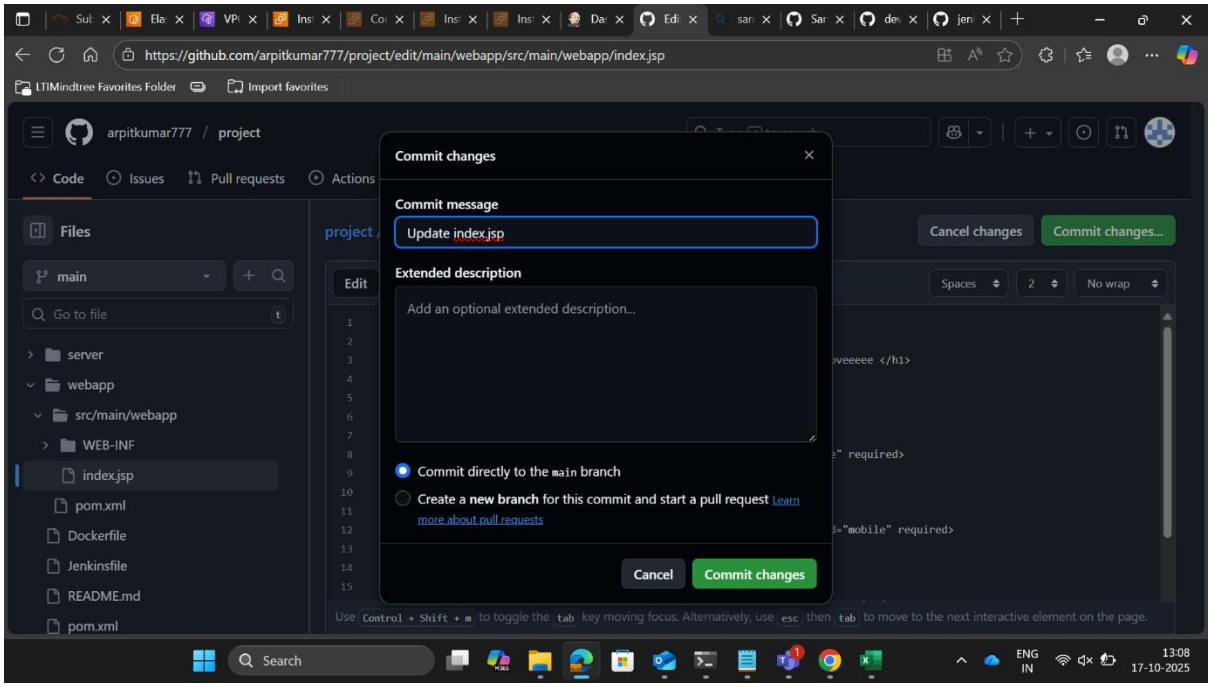

Now we are allowing GitHub hook trigger for auto build purpose if any changes are being made then there will be automatic build.



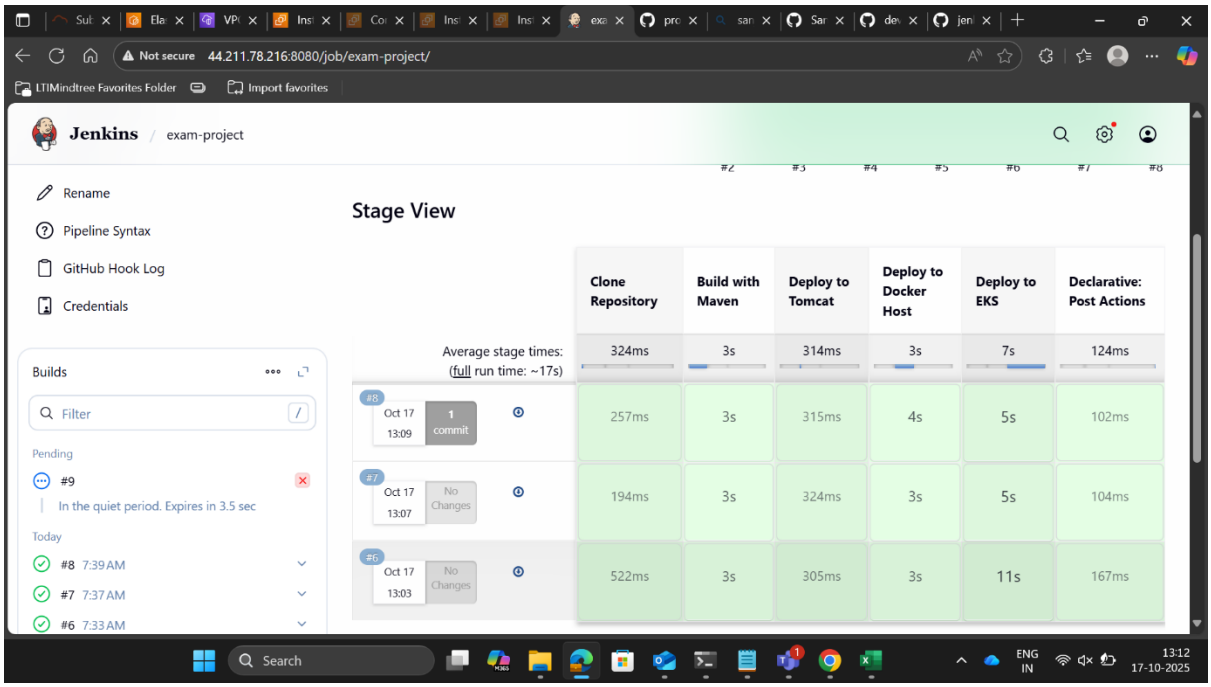
Now we are doing some changes in file form GitHub.



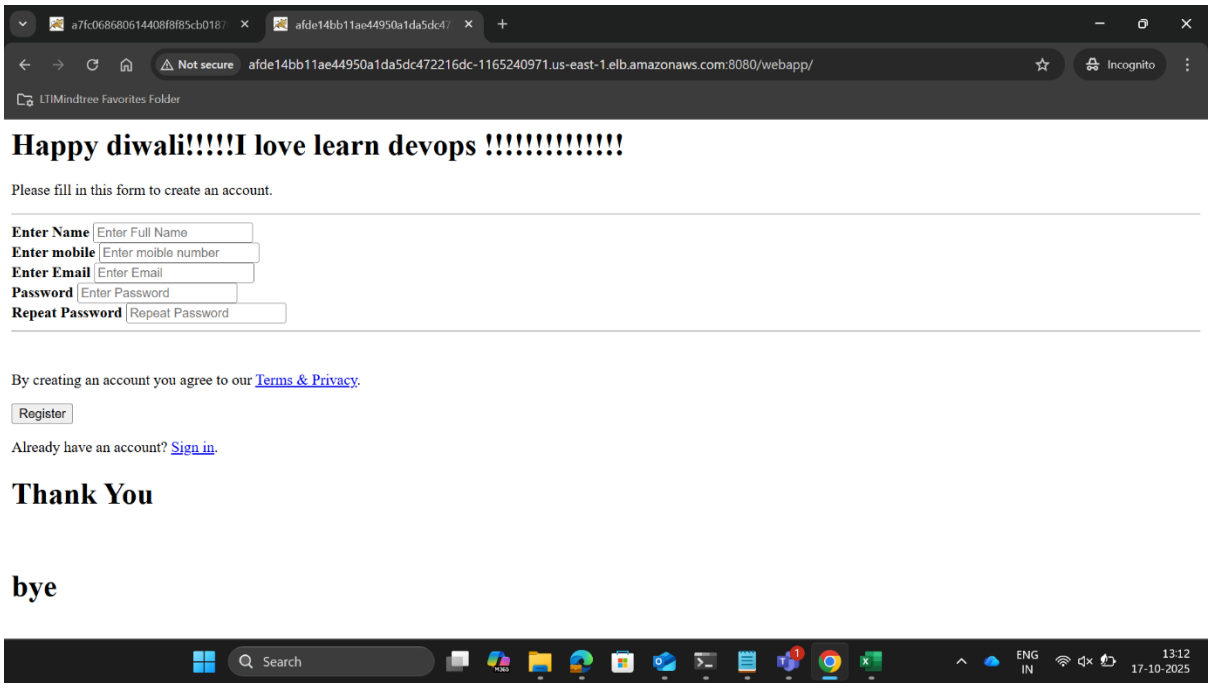

Now we are committing the changes.



We can see that there is automatic build after the changes are being committed.




We can see the changes



## Pipeline script:

```
pipeline {
  agent any
  environment {
    TOMCAT_USER = 'deployer'
    TOMCAT_PASS = 'deployer'
    TOMCAT_HOST = '3.87.231.9'
    TOMCAT_PORT = '8080'
  }
  stages {
    stage('Clone Repository') {
      steps {
        git branch: 'main', url: 'https://github.com/arpitkumar777/project.git'
      }
    }
    stage('Build with Maven') {
      steps {
        sh 'mvn clean package'
      }
    }
    stage('Deploy to Tomcat') {
      steps {
        sh '''
          curl -u $TOMCAT_USER:$TOMCAT_PASS \
          --upload-file webapp/target/webapp.war \
          "http://$TOMCAT_HOST:$TOMCAT_PORT/manager/text/deploy?path=/
webapp&update=true"
        '''
      }
    }
    stage('Deploy to Docker Host') {
      steps {
        sshPublisher(publishers: [
          sshPublisherDesc(
            configName: 'docker',
            transfers: [
```

```
                sshTransfer(
                  sourceFiles: '**/*',
                  removePrefix: '',
                  remoteDirectory: 'image',
                  execCommand: '''
                    cd image
                    aws ecr get-login-password --region us-east-1 | docker login --
username AWS --password-stdin 947390141947.dkr.ecr.us-east-
1.amazonaws.com
                    docker build -t tomcat-webapp .
                    docker tag tomcat-webapp:latest 947390141947.dkr.ecr.us-
east-1.amazonaws.com/tomcat-webapp:latest
                    docker push 947390141947.dkr.ecr.us-east-
1.amazonaws.com/tomcat-webapp:latest
                  '''
                )
              ]
            )
          ])
        }
      }
     stage('Deploy to EKS') {
   steps {
     sshPublisher(publishers: [
       sshPublisherDesc(
         configName: 'eks',
         transfers: [
           sshTransfer(
             sourceFiles: 'deployment.yaml,service.yaml',
             remoteDirectory: '', // adjust as needed
             removePrefix: '',
             execCommand: '''
               set -ex
               aws eks update-kubeconfig --region us-east-1 --name my-cluster
               kubectl delete -f deployment.yaml
               kubectl apply -f deployment.yaml
               kubectl apply -f service.yaml
               kubectl rollout status deployment/regapp-deployment
             '''
           )
         ],
```

```
            usePromotionTimestamp: false,
            verbose: true
          )
        ])
      }
    }
    }
    post {
      success {
        junit '**/target/surefire-reports/TEST-*.xml'
        archiveArtifacts artifacts: '**/target/*.war', fingerprint: true
      }
    }
}
```

NAME : ARPIT KUMAR

PS ID: 10844621