SARAN AVINASH B – 22CSR179

DEVOPS DAY 5 – Deploy Java app to minikube automated with jenkins

Deploy Java App to Minikube Automated with Jenkins

1. Overview

Automating the deployment of a Java application to Minikube using Jenkins involves building the application, creating a Docker image, pushing it to a container registry, and deploying it to Minikube using Kubernetes manifests.

---

2. Key Concepts

A. Jenkins Pipeline

Jenkins automates the CI/CD process using a declarative pipeline. The pipeline consists of multiple stages such as:

- **SCM Checkout:** Fetches code from a repository (GitHub/GitLab).
- **Build & Test:** Uses Maven (mvn package) to compile and test the Java application.
- **Docker Build & Push:** Builds a Docker image of the application and pushes it to Docker Hub.
- **Deploy to Minikube:** Uses kubectl to apply Kubernetes deployment and service files.

B. Minikube

Minikube is a lightweight Kubernetes cluster for local development and testing. It allows developers to run Kubernetes locally and deploy applications without needing a cloud-based cluster.

Commands to Start Minikube:

sh

CopyEdit

minikube start

kubectl cluster-info

kubectl get nodes

C. Docker

Docker is used to package the Java application into a container image, making it portable and easy to deploy across environments.

Dockerfile Example:

dockerfile

CopyEdit

FROM openjdk:11

COPY target/webapp.jar /app/webapp.jar

WORKDIR /app

CMD ["java", "-jar", "webapp.jar"]

D. Kubernetes Deployment

Kubernetes YAML files define how the application should be deployed inside the Minikube cluster.

Deployment YAML Example:

yaml

CopyEdit

apiVersion: apps/v1

kind: Deployment

metadata:

  name: webapp

spec:

  replicas: 1

  selector:

```
  matchLabels:
    app: webapp
 template:
  metadata:
    labels:
      app: webapp
  spec:
    containers:
      - name: webapp
        image: saranavinashb/webapp1
        ports:
          - containerPort: 8080
```

Apply Deployment:

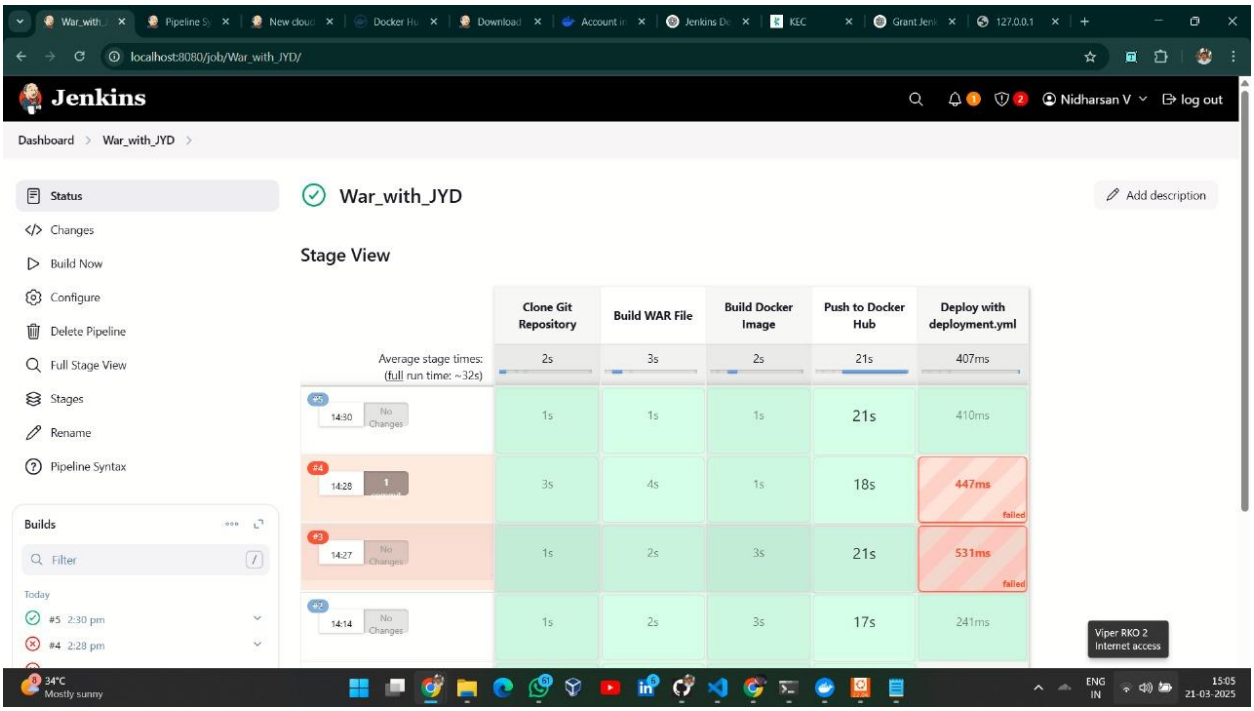
```
sh
CopyEdit
kubectl apply -f deployment.yml
kubectl get pods
```

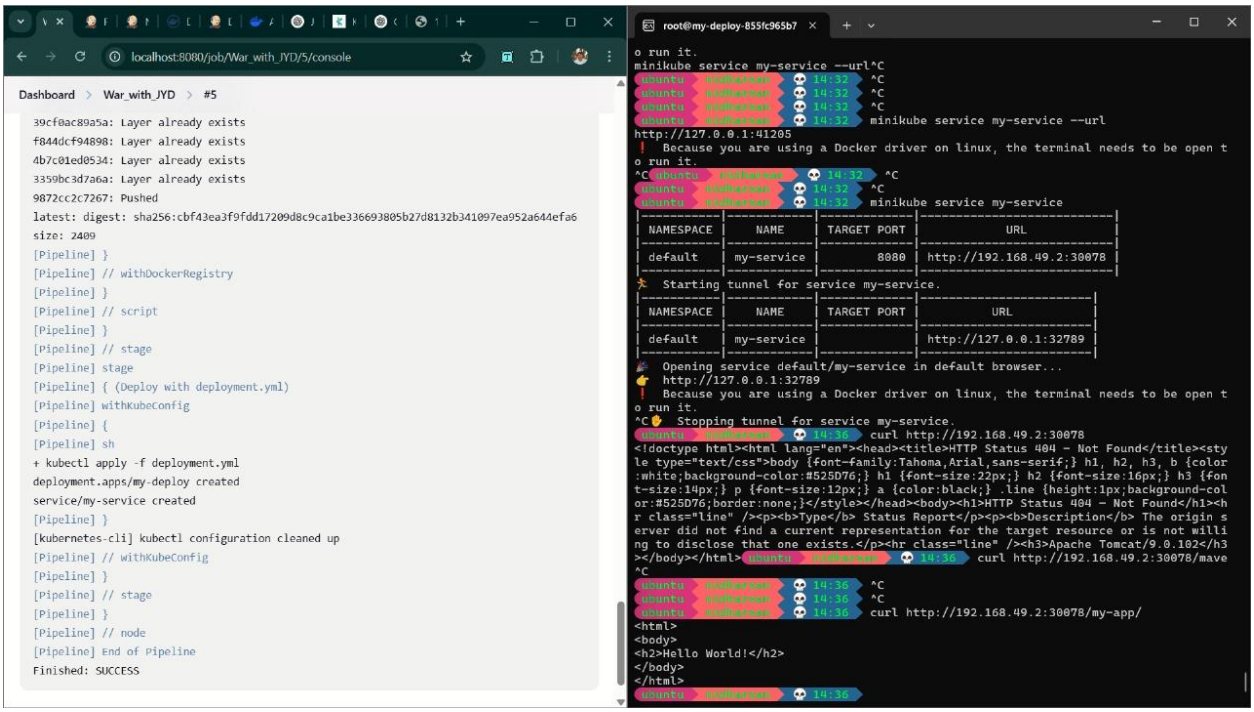
```
pipeline {
   agent any

   stages {
      stage('scm') {
         steps {
      git branch: ''
```

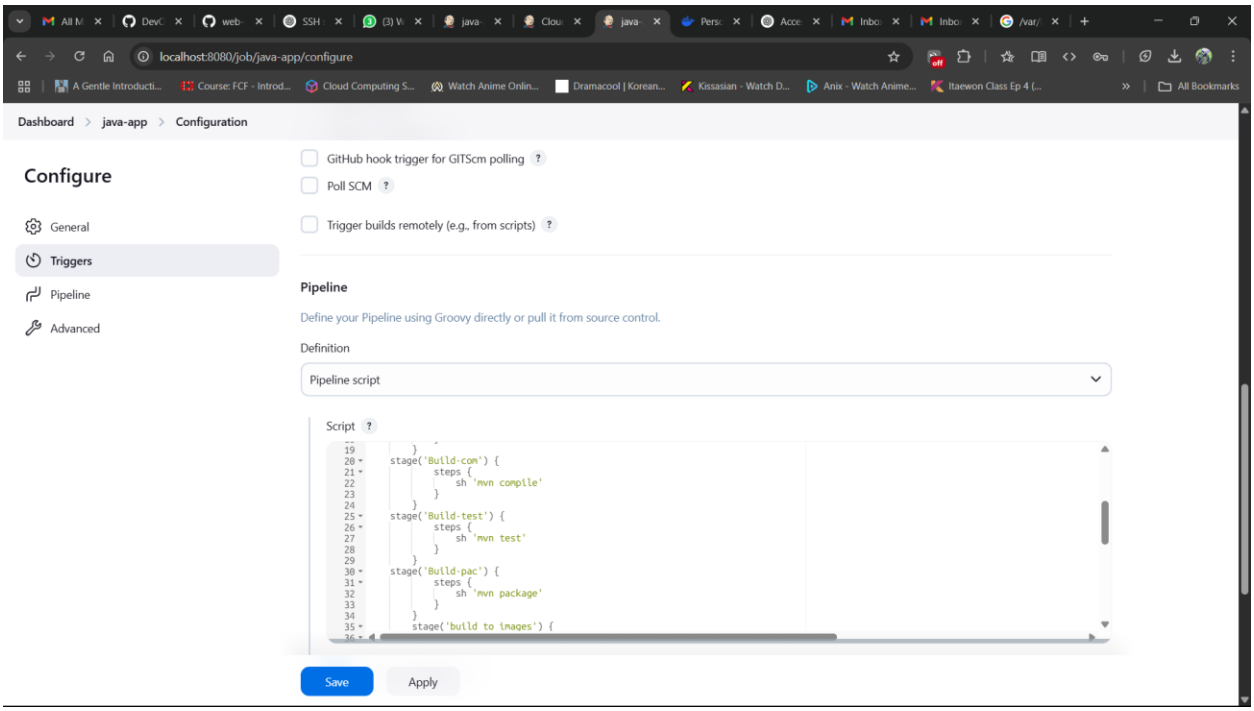
```
        }
    }
    stage('builb-clean') {
        steps {
            sh "mvn clean"
}
}
    stage('build-validate') {
        steps {
            sh "mvn validate"
}
}
    stage('build-com') {
        steps {
            sh "mvn compile"
}
}

    stage('build-test') {
        steps {
            sh "mvn test"
}
}
    stage('build-install') {
        steps {
            sh "mvn package"
```

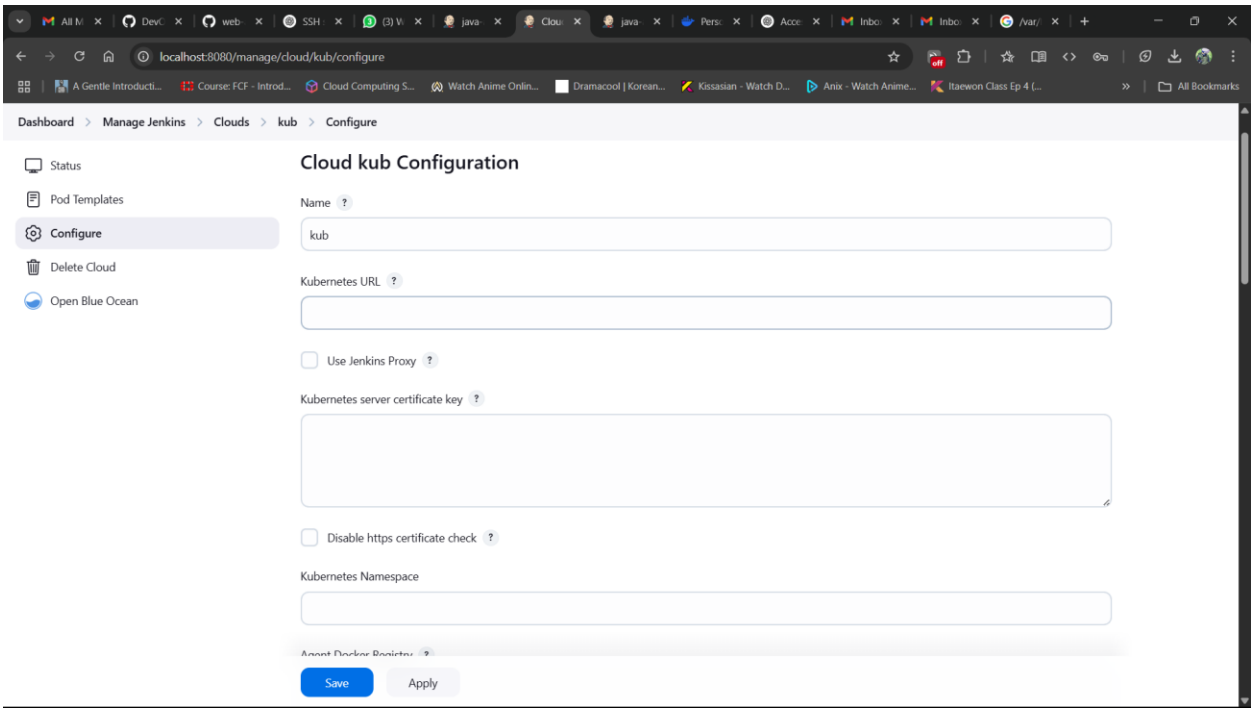
```
}
}
stage('build to images') {
        steps {
          script{
            sh 'docker build -t  .'
          }
   }
}
stage('push to hub') {
        steps {
          script{
           withDockerRegistry(credentialsId: 'Docker_cred', url: 'https://index.docker.io/v1/') {
            sh 'docker push '
          }
        }
        }
}
      stage('Deploy App') {
        steps {
           withKubeConfig(caCertificate: '', clusterName: 'minikube', contextName: 'minikube', credentialsId: 'mukubeconfig_011', namespace: '', restrictKubeConfigAccess: false, serverUrl: 'https://192.168.49.2:8443') {
            sh 'kubectl apply -f deployment.yml --validate=false'
        }
      }
      }
```

```
    stage('Test') {
      steps {
        withKubeConfig(caCertificate: '', clusterName: 'minikube', contextName: 'minikube', credentialsId: 'mukubeconfig_011', namespace: '', restrictKubeConfigAccess: false, serverUrl: 'https://192.168.49.2:8443') {

        sh 'minikube service my-service --url | xargs curl'
    }
}
}
}
}
```

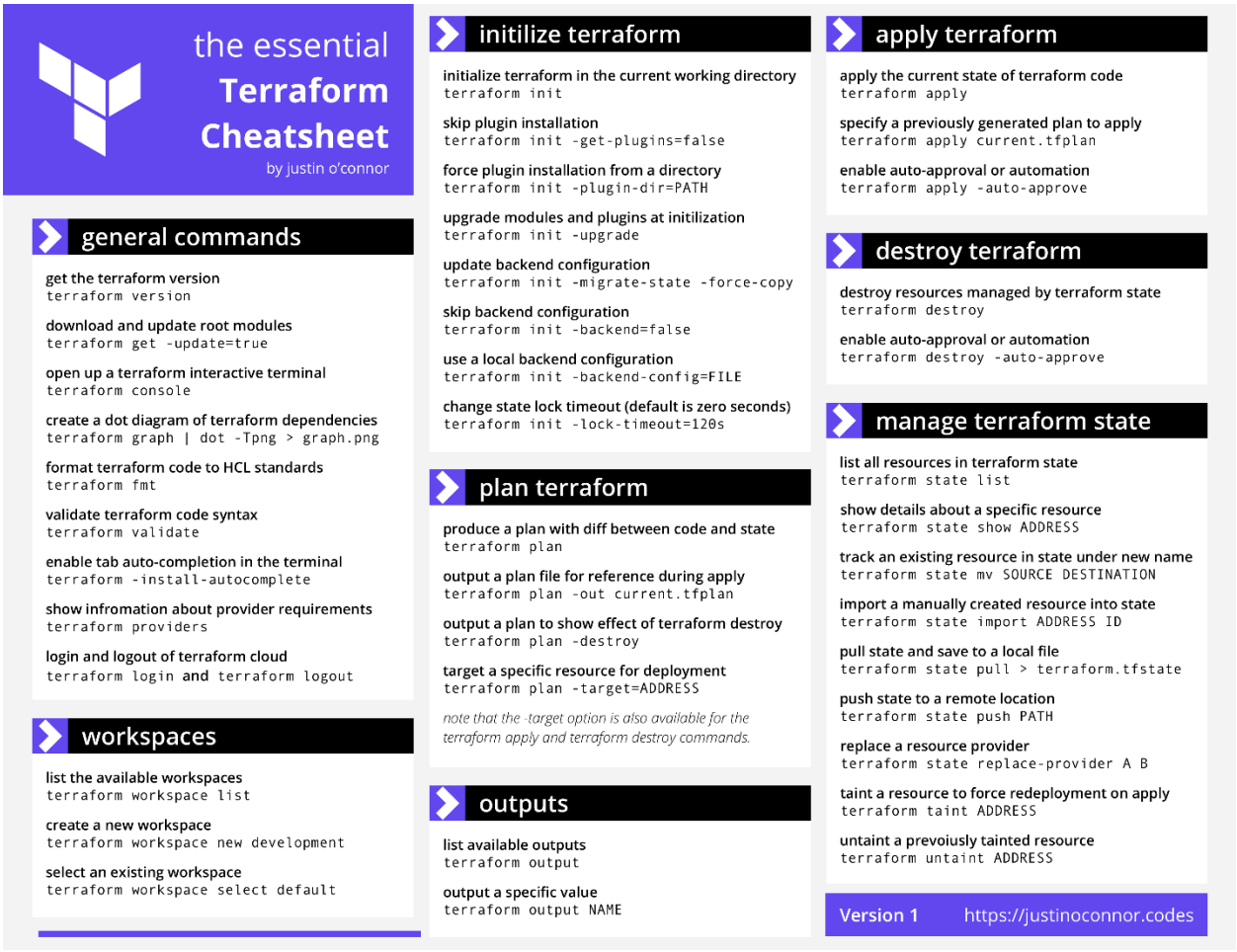

Jenkins
Nidharsan V
log out
Dashboard > War_with_JYD >
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Stages
Rename
Pipeline Syntax
War_with_JYD
Add description
Stage View
Clone Git Repository
Build WAR File
Build Docker Image
Push to Docker Hub
Deploy with deployment.yml
Average stage times:
(full run time: ~32s)
2s
3s
2s
21s
407ms
#5 14:30 No Changes
1s
1s
1s
21s
410ms
#4 14:28
3s
4s
1s
18s
447ms
failed
#3 14:27 No Changes
1s
2s
3s
21s
531ms
failed
#2 14:14 No Changes
1s
2s
3s
17s
241ms
Builds
Filter
Today
#5 2:30 pm
#4 2:28 pm
Viper RKO 2
Internet access
34°C
Mostly sunny
ENG
IN
15:05
21-03-2025


localhost:8080/job/War_with_JYD/5/console
Dashboard > War_with_JYD > #5
39cf0ac89a5a: Layer already exists
f844dcf94898: Layer already exists
4b7c01ed0534: Layer already exists
3359bc3d7a6a: Layer already exists
9872cc2c7267: Pushed
latest: digest: sha256:cbf43ea3f9fdd17209d8c9ca1be336693805b27d8132b341097ea952a644efa6
size: 2409
[Pipeline] }
[Pipeline] // withDockerRegistry
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy with deployment.yml)
[Pipeline] withKubeConfig
[Pipeline] {
[Pipeline] sh
+ kubectl apply -f deployment.yml
deployment.apps/my-deploy created
service/my-service created
[Pipeline] }
[kubernetes-cli] kubectl configuration cleaned up
[Pipeline] // withKubeConfig
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
root@my-deploy-855fc965b7
o run it.
minikube service my-service --url^C
minikube service my-service --url
http://127.0.0.1:41205
Because you are using a Docker driver on linux, the terminal needs to be open t
o run it.
minikube service my-service
NAMESPACE | NAME | TARGET PORT | URL
default | my-service | 8080 | http://192.168.49.2:30078
Starting tunnel for service my-service.
NAMESPACE | NAME | TARGET PORT | URL
default | my-service | | http://127.0.0.1:32789
Opening service default/my-service in default browser...
http://127.0.0.1:32789
Because you are using a Docker driver on linux, the terminal needs to be open t
o run it.
Stopping tunnel for service my-service.
curl http://192.168.49.2:30078
<!doctype html><html lang="en"><head><title>HTTP Status 404 – Not Found</title><sty
le type="text/css">body {font-family:Tahoma,Arial,sans-serif;} h1, h2, h3, b {color
:white;background-color:#525D76;} h1 {font-size:22px;} h2 {font-size:16px;} h3 {fon
t-size:14px;} p {font-size:12px;} a {color:black;} .line {height:1px;background-col
or:#525D76;border:none;}</style></head><body><h1>HTTP Status 404 – Not Found</h1><h
r class="line" /> Type Status Report Description The origin s
erver did not find a current representation for the target resource or is not willi
ng to disclose that one exists. <hr class="line" /><h3>Apache Tomcat/9.0.102</h3
></body></html> curl http://192.168.49.2:30078/mave
curl http://192.168.49.2:30078/my-app/
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>


localhost:8080/job/java-app/configure
Dashboard > java-app > Configuration
Configure
General
Triggers
Pipeline
Advanced
GitHub hook trigger for GITScm polling
Poll SCM
Trigger builds remotely (e.g., from scripts)
Pipeline
Define your Pipeline using Groovy directly or pull it from source control.
Definition
Pipeline script
Script
stage('Build-com') {
sh 'mvn compile'
stage('Build-test') {
sh 'mvn test'
stage('Build-pac') {
sh 'mvn package'
stage('build to images') {
Save
Apply


localhost:8080/manage/cloud/kub/configure
Dashboard > Manage Jenkins > Clouds > kub > Configure
Status
Pod Templates
Configure
Delete Cloud
Open Blue Ocean
Cloud kub Configuration
Name
kub
Kubernetes URL
Use Jenkins Proxy
Kubernetes server certificate key
Disable https certificate check
Kubernetes Namespace
Save
Apply

Terraform




```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.92.0"
    }
  }
```

```
}

provider "aws" {
   region = "us-east-1"
}

resource "aws_vpc" "myvpc" {
  cidr_block       = "10.0.0.0/16"

  tags = {
    Name = "demovpc"
  }
}

resource "aws_subnet" "pubsub" {
  vpc_id     = aws_vpc.myvpc.id
  cidr_block = "10.0.1.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "sn1"
  }
}

resource "aws_subnet" "pub_sub" {
  vpc_id     = aws_vpc.myvpc.id
```

```
  cidr_block = "10.0.2.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "sn1"
  }
}

resource "aws_subnet" "prisub" {
  vpc_id     = aws_vpc.myvpc.id
  cidr_block = "10.0.3.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "sn1"
  }
}

resource "aws_subnet" "pri_sub" {
  vpc_id     = aws_vpc.myvpc.id
  cidr_block = "10.0.4.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "sn1"
  }
```

```
}

resource "aws_internet_gateway" "tfigw" {
 vpc_id = aws_vpc.myvpc.id

 tags = {
  Name = "tfigw"
 }
}

resource "aws_route_table" "tfpubrt" {
 vpc_id = aws_vpc.myvpc.id

 route {
  cidr_block = "0.0.0.0/0"
  gateway_id = aws_internet_gateway.tfigw.id
 }

 tags = {
  Name = "tfpublicroute"
 }
}

resource "aws_route_table_association" "pubsn1" {
 subnet_id      = aws_subnet.pubsub.id
 route_table_id = aws_route_table.tfpubrt.id
```

```
}

resource "aws_route_table_association" "pubsn2" {
 subnet_id      = aws_subnet.pub_sub.id
 route_table_id = aws_route_table.tfpubrt.id
}

resource "aws_eip" "tfeip" {
 domain   = "vpc"
}

resource "aws_nat_gateway" "tfnat" {
 allocation_id = aws_eip.tfeip.id
 subnet_id     = aws_subnet.pub_sub.id

 tags = {
   Name = "gw NAT"
 }
}

resource "aws_route_table" "tfprirt" {
 vpc_id = aws_vpc.myvpc.id

 route {
   cidr_block = "0.0.0.0/0"
   gateway_id = aws_nat_gateway.tfnat.id
```

```
  }

  tags = {
    Name = "tfprivateroute"
  }
}

resource "aws_route_table_association" "prisn3" {
  subnet_id      = aws_subnet.prisub.id
  route_table_id = aws_route_table.tfprirt.id
}
resource "aws_route_table_association" "prisn4" {
  subnet_id      = aws_subnet.pri_sub.id
  route_table_id = aws_route_table.tfprirt.id
}

resource "aws_security_group" "allow_tfsg" {
  name        = "allow_tfsg"
  description = "Allow TLS inbound traffic"
  vpc_id      = aws_vpc.myvpc.id

  ingress {
    description      = "HTTPS "
    from_port        = 443
    to_port          = 443
    protocol         = "tcp"
```

```
  cidr_blocks      = ["0.0.0.0/0"]
}
 ingress {
  description      = "HTTP "
  from_port        = 80
  to_port          = 80
  protocol         = "tcp"
  cidr_blocks      = ["0.0.0.0/0"]
}
 ingress {
  description      = "SSH"
  from_port        = 22
  to_port          = 22
  protocol         = "tcp"
  cidr_blocks      = ["0.0.0.0/0"]
}

egress {
  from_port        = 0
  to_port          = 0
  protocol         = "-1"
  cidr_blocks      = ["0.0.0.0/0"]
}

tags = {
  Name = "TfsecurityGroup"
```

```
  }
}

resource "aws_instance" "pub_ins" {
  ami                       = "ami-0fc5d935ebf8bc3bc"
  instance_type             = "t2.micro"
  subnet_id                 = aws_subnet.pub_sub.id
  vpc_security_group_ids    = [aws_security_group.allow_tfsg.id]
 key_name                   = "saran"
 associate_public_ip_address   =  "true"
}

#terraform init
#terraform validate
#terraform plan
#terraform apply
#terraform destroy
```