INT 334 Enterprise Application Automation

# CA – 1

Name : Shubhansu Kumar Singh

Reg No : 12104991  
Section : KO308  
Roll : B59  
Date : 19-02-2025

Practical : Create a sample Kubernetes deployment establishing a master -slave connection.

Create Shubhansu-Master EC2 instance

A screenshot of a computer

AI-generated content may be incorrect.

Create Singh-Slave EC2 instance

A screenshot of a computer

AI-generated content may be incorrect.

Initialize Kubernetes cluster in master node

A screenshot of a computer

AI-generated content may be incorrect.

Join slave node to the cluster :   
A screenshot of a computer

AI-generated content may be incorrect.

Check node status in master node : Cluster ready

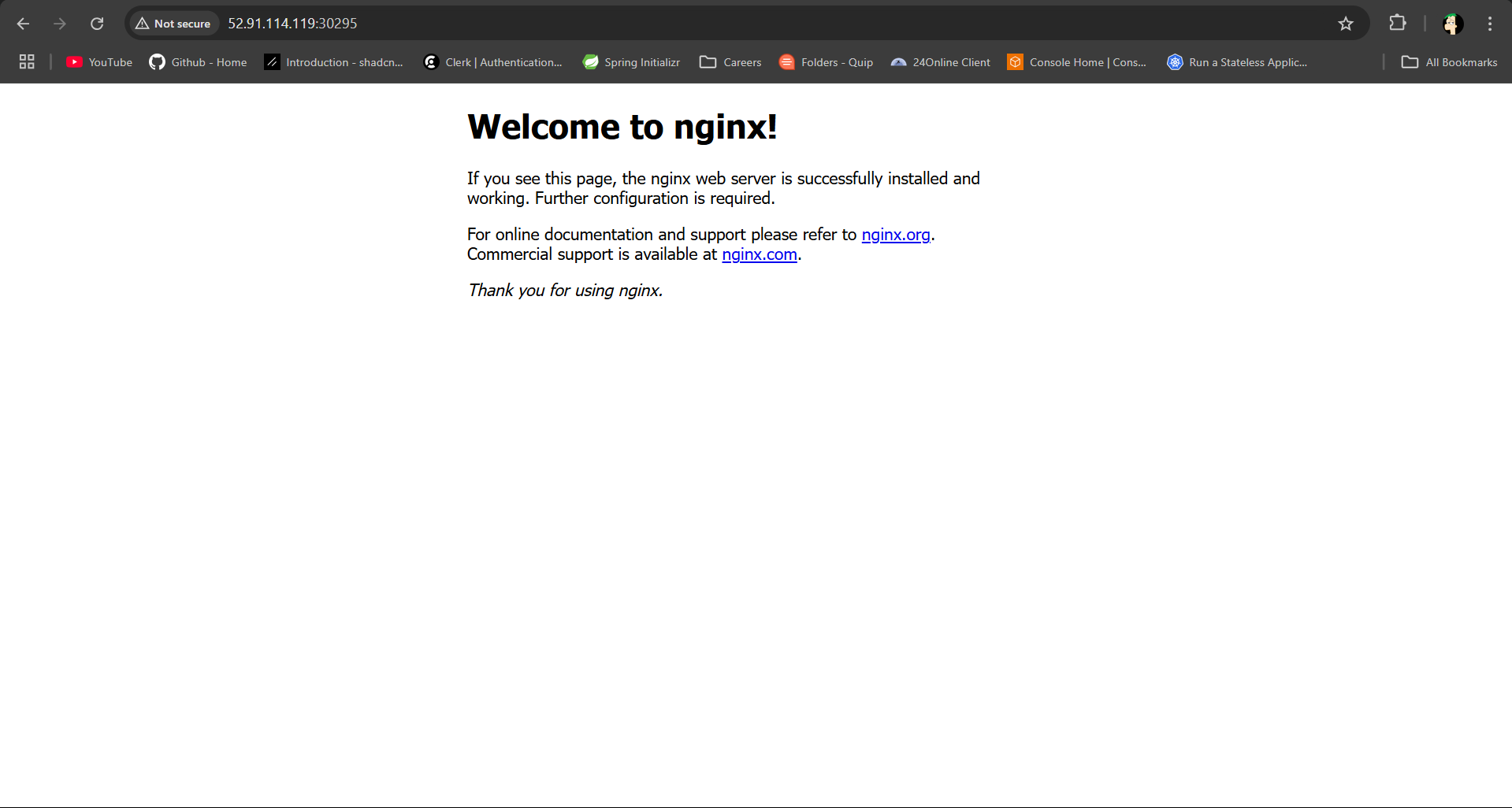
A screenshot of a computer program

AI-generated content may be incorrect.

Create a sample deployment and check status

A screenshot of a computer

AI-generated content may be incorrect.



The above mentioned screen shot shows the successful deployment of sample service in Kubernetes service.

The following commands were executed for this practical.

Setup MASTER-NODE Connection

#!/bin/bash

# Function to log messages

log() {

    echo "[INFO] *$1*"

}

# Set hostname (Modify as needed)

NODE\_TYPE=*$1*

if [ "$NODE\_TYPE" == "master" ]; then

    sudo hostnamectl set-hostname master

elif [ "$NODE\_TYPE" == "worker" ]; then

    sudo hostnamectl set-hostname worker

else

    echo "Usage: *$0* [master|worker]"

    exit 1

fi

# Update system

log "Updating system..."

sudo apt-get update && sudo apt-get upgrade -y

# Disable swap

log "Disabling swap..."

sudo swapoff -a

# Load necessary kernel modules

log "Loading required kernel modules..."

cat << EOF | sudo tee /etc/modules-load.d/k8s.conf

overlay

br\_netfilter

EOF

sudo modprobe overlay

sudo modprobe br\_netfilter

# Set sysctl parameters

log "Configuring networking..."

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf

net.bridge.bridge-nf-call-iptables  = 1

net.bridge.bridge-nf-call-ip6tables = 1

net.ipv4.ip\_forward                 = 1

EOF

sudo sysctl --system

lsmod | grep br\_netfilter

lsmod | grep overlay

# Install container runtime (containerd)

log "Installing container runtime..."

sudo apt-get update

sudo apt-get install -y ca-certificates curl

# Add Docker GPG key and repository

log "Adding Docker repository..."

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu $(. /etc/os-release && echo \"$VERSION\_CODENAME\") stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install -y containerd.io

# Configure containerd

log "Configuring containerd..."

containerd config default | sed -e 's/SystemdCgroup = false/SystemdCgroup = true/' -e 's/sandbox\_image = "registry.k8s.io\/pause:3.6"/sandbox\_image = "registry.k8s.io\/pause:3.9"/' | sudo tee /etc/containerd/config.toml

sudo systemctl restart containerd

sudo systemctl status containerd --no-pager

# Install Kubernetes packages

log "Installing Kubernetes components..."

sudo apt-get update

sudo apt-get install -y apt-transport-https ca-certificates curl gpg

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

# Initialize the Kubernetes cluster (Only on master node)

if [ "$NODE\_TYPE" == "master" ]; then

    log "Initializing Kubernetes master node..."

    sudo kubeadm init

    mkdir -p "$HOME/.kube"

    sudo cp -i /etc/kubernetes/admin.conf "$HOME/.kube/config"

    sudo chown "$(id -u)":"$(id -g)" "$HOME/.kube/config"

    # Apply Calico network plugin

    log "Applying Calico networking..."

    kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.26.0/manifests/calico.yaml

    log "Master node setup complete!"

    log "To join worker nodes, use the following command:"

    kubeadm token create --print-join-command > join-command.sh

    chmod +x join-command.sh

    log "Run 'cat join-command.sh' to see the join command."

elif [ "$NODE\_TYPE" == "worker" ]; then

    log "Resetting Kubernetes on worker node..."

    sudo kubeadm reset --force

    log "Joining the Kubernetes cluster..."

    # Run the join command (manual step)

    log "Run the join command from the master node."

fi

# For Master

# chmod +x setup\_k8s.sh

# ./setup\_k8s.sh master

# cat join-command.sh

# For Worker

# chmod +x setup\_k8s.sh

# ./setup\_k8s.sh worker

# Copy the join command from the master node and run it manually on the worker node.

Create Deployment

#!/bin/bash

# Function to log messages with spacing

log() {

echo -e "\n[INFO] $1\n"

}

# Cleanup option - Placed at the beginning to avoid unnecessary execution

if [ "$1" == "cleanup" ]; then

log "Deleting the Nginx deployment..."

kubectl delete deployment nginx-deployment

log "Deleting the Nginx service..."

kubectl delete service nginx

log "Cleanup complete!"

exit 0

fi

# Define YAML file name and URL

YAML\_FILE="nginx-deployment.yaml"

YAML\_URL="https://raw.githubusercontent.com/kubernetes/website/main/content/en/examples/controllers/nginx-deployment.yaml"

# Download YAML file

log "Downloading Nginx deployment YAML..."

curl -s -o "$YAML\_FILE" "$YAML\_URL"

if [ ! -f "$YAML\_FILE" ]; then

echo -e "\n[ERROR] Failed to download $YAML\_FILE. Exiting.\n"

exit 1

fi

# Apply the YAML configuration

log "Applying the Nginx deployment..."

kubectl apply -f "$YAML\_FILE"

# Wait for deployment to be ready

log "Waiting for the deployment to be ready..."

kubectl rollout status deployment/nginx-deployment

# Create a NodePort service for Nginx

log "Creating a NodePort service for Nginx..."

kubectl create service nodeport nginx --tcp=80:80

# Display service information

log "Fetching details of the created Nginx service..."

kubectl get svc nginx

# Display deployment and cluster info

log "Listing all running pods..."

kubectl get pods

sleep 1

log "Listing all deployments..."

kubectl get deployment

sleep 1

log "Listing all services..."

kubectl get service

sleep 1

log "Fetching cluster information..."

kubectl cluster-info

sleep 1

log "Listing all pods across all namespaces..."

kubectl get pods --all-namespaces

log "Nginx deployment setup complete!"