**Runner-Deployment**

**Prerequisites:**

1.Virtual Machine with following tools and packages installed.

Docker, Mini Kube, Kubectl ,Helm,Cert-manager

2.GitHub account with write access to any repository.

**1.1Docker-Installation**

sudo apt-get update

sudo apt-get install ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

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 \

$(. /etc/os-release && echo "${UBUNTU\_CODENAME:-$VERSION\_CODENAME}") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

sudo systemctl enable docker

sudo systemctl start docker

sudo systemctl status docker

After Installing thedocker, check docker status using the below command.

sudo systemctl status docker

**A computer screen with text on it

AI-generated content may be incorrect.**

**1.2 MiniKube Installation**

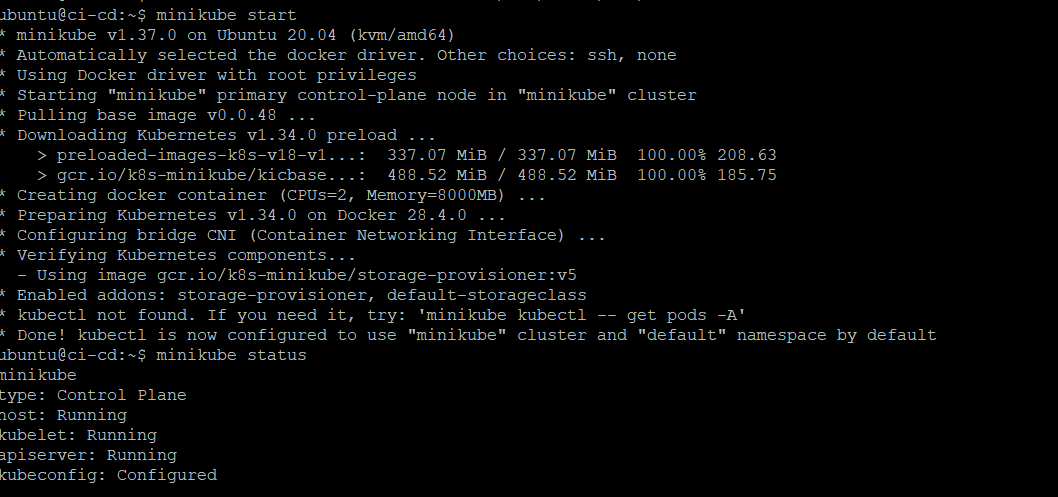
curl -LO https://github.com/kubernetes/minikube/releases/latest/download/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64

minikube start

#check the status of minikube

minikube status

****

**1.3 Install Kubectl**

curl -LO [https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl](https://dl.k8s.io/release/$(curl%20-L%20-s%20https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl)

curl -LO [https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256](https://dl.k8s.io/release/$(curl%20-L%20-s%20https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl.sha256)

sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl

chmod +x kubectl

mkdir -p ~/.local/bin

mv ./kubectl ~/.local/bin/kubectl

**1.4 Install Helm Charts**

curl -fsSL -o get\_helm.sh <https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3>

chmod 700 get\_helm.sh

./get\_helm.sh

**1.5 Install Cert Manager**

helm repo add jetstack https://charts.jetstack.io --force-update

helm install cert-manager jetstack/cert-manager --namespace cert-manager --create-namespace --version v1.15.1 --set crds.enabled=true

Verify helm installation by checking the pods in cert-manager namespace

**A screenshot of a computer screen

AI-generated content may be incorrect.**

**2.Create a GitHub personal access Token.**

Login to GitHub account using username and password.

Click on settings->developer settings-> personal access token

Create a new personal access token by giving workflow access and set an expiry date for the token.

**A screenshot of a computer

AI-generated content may be incorrect.**

**3.Generation of Self-Hosted-Runners for repositories.**

#create a Kubernetes namespace for runner stack

kubectl create ns actions-runner-system

create a Kubernetes secret along with the github personal access token that was created earlier

kubectl create secret generic controller-manager -n actions-runner-system --from-literal=github\_token=<GITHUB-ACCESS-TOKEN>

**3.1Install action-runner-controller using helm**

# Adding Helm Repo of ARC  
helm repo add actions-runner-controller https://actions-runner-controller.github.io/actions-runner-controller  
  
# Updating Helm Repo  
helm repo update  
  
# Installing ARC  
helm upgrade --install --namespace actions-runner-system --create-namespace --wait actions-runner-controller actions-runner-controller/actions-runner-controller --set syncPeriod=1m

**A black screen with white text

AI-generated content may be incorrect.**

**3.2 Create a yaml file for runner deployment.**

sudo vi /home/ubuntu/RunnerDeployment.yaml

apiVersion: actions.summerwind.dev/v1alpha1

kind: RunnerDeployment

metadata:

name: k8s-adminui-runner

namespace: actions-runner-system

spec:

replicas: 1

template:

spec:

repository: RiteProductsPlatform/ConvertRite.RiteAdminUI

labels:

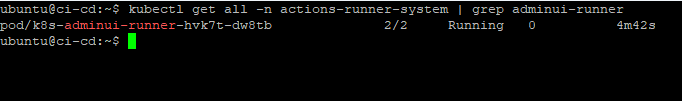
- "AdminUI\_Runner"

**Now create the runner by running the above yaml file using Kubernetes**

Kubectl apply -f/home/ubuntu/RunnerDeployment.yaml

**Verify the runner deployment using below command**

kubectl get all -n actions-runner-system | grep adminui-runner



Now login to the Github and go to the repository for runner verification.

Go to Repository->settings->actions-> runners

A screenshot of a computer

AI-generated content may be incorrect.

**3.3 Create the first workflow**

Go to repository -> actions->new workflow

A screenshot of a computer

AI-generated content may be incorrect.

After creating the workflow, specify the runner label at **runs-on** section in the workflow

A white background with blue text

AI-generated content may be incorrect.

When ever a push happens to the specific repository, the ci/cd pipeline will be automatically triggered and performs all the steps that are mentioned in the yaml file.

We can see the status of the yaml file after it’s completion.

