**A**

**Project Report**

**is**

**Submitted to the completion of Epam-COE-DevOPS**

**Training Program**



**Submitted By:**

**Nitika Raghwa**

**Junior Software Engineer-Epam Hyderabad**

**PROJECT-1**

**Following is the documentation of project that covers following objectives:**

1. Three Tier web application using docker and Kubernetes
2. Infrastructure as Code Using Terraform (Modules)
3. Configuration Management using Ansible (Roles)
4. Application code management using Git
5. Building CI/CD pipeline to deploy a new version of Application (Jenkins)
6. Building Monitoring for application

**I have used Virtual Machines for three tier application.**

### Step 1: Install Docker to takes away repetitive, mundane configuration tasks and for fast development lifecycle.

### Step 2: Start and enable the docker using systemctl

Text

Description automatically generated

# **Kubenetes Installation and cluster creation**

Step3: Install containerd: **container runtime that manages the lifecycle of a container on a physical or virtual machine (a host)**. It is a daemon process that creates, starts, stops, and destroys containers.

Step 4: Start and enable the containerd if required.

Text

Description automatically generated with medium confidence

**Install packages**

**sudo apt-get install -y kubelet kubeadm kubectl**

**Prevent auto updates for kube packages**

**sudo apt-mark hold kubelet kubeadm kubectl**

**done**

**Bootstrapping cluster**

**sudo kubeadm init --pod-network-cidr=10.244.0.0/16**

Step 5: Check the status of kubelet: kubelet is **the primary "node agent" that runs on each node**. It can register the node with the apiserver using one of: the hostname; a flag to override the hostname; or specific logic for a cloud provider.

Text

Description automatically generated with low confidence

Check for running containers:

A picture containing calendar

Description automatically generated

#### K8s manifest files

* mongo-config.yaml
* mongo-secret.yaml
* mongo.yaml
* webapp.yaml

**mongo-config.yaml File** : object configuration file / configuration file: **A file that defines the configuration for a Kubernetes object**.

**Mongo-secret.yaml File** : Kubernetes Secrets are secure objects which store sensitive data, such as passwords, OAuth tokens, and SSH keys, etc. with encryption in your clusters. Using Secrets **gives you more flexibility in a Pod Life cycle definition and control over how sensitive data is used**.

**Mongo.yaml and webapp.yaml file:** These files contain Deployment and service creation code for mongo DB and my website

A Deployment provides declarative updates for [Pods](https://kubernetes.io/docs/concepts/workloads/pods/) and [ReplicaSets](https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/" \o "" \t "_blank).

You describe a desired state in a Deployment, and the Deployment [Controller](https://kubernetes.io/docs/concepts/architecture/controller/) changes the actual state to the desired state at a controlled rate.

A Service in Kubernetes is a REST object, similar to a Pod. Like all of the REST objects, you can POST a Service definition to the API server to create a new instance.

##### Get basic info about k8s components

kubectl get node

kubectl get pod

kubectl get svc

kubectl get all

Graphical user interface

Description automatically generated

**Final Output:**

Graphical user interface, website

Description automatically generated

**After Making some modifications in frontend the changes will be stored in Database and will get reflected on the page:**

Graphical user interface

Description automatically generated

**2) Infrastructure as Code Using Terraform (Modules)**

The code will be shared in the git repo.

Text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generated

**Application code management using Git**

Git is **an Open-Source Distributed Version Control System**.

Now I will be pushing my code to my GitHub repository.

Text

Description automatically generated

First Install Git in your system:

Text

Description automatically generated

Check the git –version

Text

Description automatically generated

The git init command Initializes a new Git repository. It can be used to convert an existing, unversioned project to a Git repository or initialize a new, empty repository.

Text

Description automatically generated with medium confidence

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Text

Description automatically generated

Git add: Moves changes from the working directory to the staging area.



Text

Description automatically generated

Git commit: Takes the staged snapshot and commits it to the project history.

Text

Description automatically generated

Git push: It lets you move a local branch to another repository, which serves as a convenient way to publish contributions.

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Configuration Management using Ansible (Roles)

Building CI/CD pipeline to deploy a new version of Application (Jenkins)

Building Monitoring for application

1. Create three Instances using AWS:
2. Ansible
3. Jenkins
4. Docker

Graphical user interface, application

Description automatically generated

Ansible Server

Text

Description automatically generated

Docker server

Graphical user interface, text

Description automatically generated

1. On Jenkins Server:
2. Install java

Yum install java

1. Install Jenkins

* **sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo**
* **sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key**
* **yum install Jenkins**

1. **Run**

systemctl starts jenkins

systemctl enable Jenkins

Graphical user interface, text, application, email

Description automatically generated

**Add the Jenkins url to webhooks in git to automatically trigger the build of a Jenkins project in response to a commit pushed to a Git repository**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Add the Git Url in source code management for making the integration

Graphical user interface, text, application, email, Teams

Description automatically generated

Graphical user interface, text, application, email, Teams

Description automatically generated

Sync the Jenkins and ansible server for the transfer of Dockerfile from Jenkins to ansible server with the help of pipeline.

Graphical user interface, text, application, email

Description automatically generated

Established the passwordless automatic connection between Jenkins to Ansible and from Ansible to Docker server.

Build – stage1 (Docker file transferred to ansible server by using pipeline)

Graphical user interface, text, application

Description automatically generated

A picture containing text

Description automatically generated

Below code is used to build the image in ansible server using the dockerfile.

Graphical user interface, text, application, email

Description automatically generated

After the post build the container will be created in Docker server with the help of code written in the playbook.

Graphical user interface, text, application, email, Teams

Description automatically generated

Stage 2: Image build on docker server

Graphical user interface, text

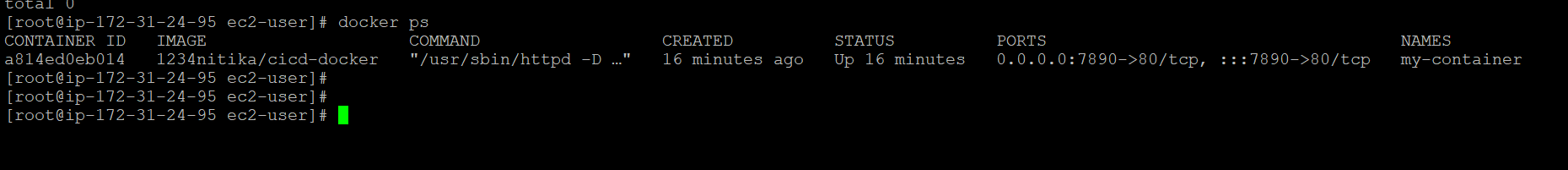
Description automatically generated

Playbook for the creation of new container using docker image and for the deletion of old containers if any:

Text

Description automatically generated

Stage 3: Docker server (Container created)



**Final output:**

After hitting the public ip of Docker Server with port 7890 we can see the website running on browser.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, website

Description automatically generated

6. Building Monitoring using CloudWatch

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

After Increasing CPU Utilization

A screenshot of a computer

Description automatically generated