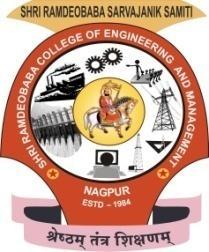
****

**SHRI RAMDEOBABA COLLEGE OF ENGINEERING AND MANAGEMENT**

**Department of Computer Science & Engineering**

**Session: 2021-22**

**CLOUD COMPUTING (CST452-4)**

**ASSIGNMENT**

**VII Semester, B.E.**

**Shift-I**

**Group Members**

**Tanisha Agrawal (A-13)**

**Bhagyashree Mahalle (A-02)**

**Sanskruti Talmale(A-06)**

**Course Co-ordinator**

**Prof.Heena Agrawal**

## TABLE OF CONTENTS

#### Page No.

1. [**AIM 1**](#_heading=h.30j0zll)
2. [**ABSTRACT 4**](#_heading=h.1fob9te)
3. [**TOOLS AND LANGUAGE USED 7**](#_heading=h.3znysh7)
4. [**INTRODUCTION 12**](#_heading=h.2et92p0)
5. **IMPLEMENTATION**

#### 

1. **AIM:**

Kubernetes and the hybrid cloud with Skupper.

1. **ABSTRACT**:

We performed hybrid cloud on kubernetes using skupper and deployed microservices. This project demonstrates how to deploy a set of http servers across multiple clusters and observe anycast application routing over a Virtual Application Network.We will deploy http servers to both a public and a private cluster. We will also create http clients that will access the http servers via the same address.

1. **TOOLS AND LANGUAGES USED:**

* Requirements:

The requirements for the setup included 2 or more CPUs, 2 GB of spare space, Free disc space of 20 GB ,Connection to the internet, Docker, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, or VMWare are examples of container or virtual machine managers.

* Skupper
* Kubernetes

1. **INTRODUCTION:**

Skupper:

Skupper is a service connection at layer 7. It allows for secure communication between Kubernetes clusters, additionally making it possible without the need for VPNs or specific firewall rules. Your application may span various cloud providers, data centres, and geographies using Skupper.

Owing to its advantages, skupper is preferred since it is simple to set up, secure by design, allows to connect anywhere and enables smart routing

With Skupper, there are no modifications that need to be made to your existing application. Additionally, there is no need for administrator rights. Skupper helps interact with other clusters without exposing service ports to the internet, mutual TLS protects inter-cluster communication; Skupper does not have any special permissions. It allows edge-to-edge connection, multicloud, and hybrid cloud; without VPNs, secure access to private cloud services from the public cloud, dynamically add and delete new clusters

Skupper lets us load balancing in real time dependent on service capacity, traffic routing that is cost- and location-aware, in the event of a network breakdown, redundant routes ensure high availability.

Minikube(Kubernetes):

Minikube is a local Kubernetes that focuses on making Kubernetes easy to understand and build for Kubernetes is only a single command away if you have a Docker (or similarly comparable) container or a Virtual Machine environment: minikube start. Furthermore, the process included the installation and setup procedure, proceeding with starting the cluster, interacting with the cluster, applications deployment and cluster management.

Linux

1. **IMPLEMENTATION:**

**Step 1: Install the Skupper command-line tool in your environment**

The skupper command-line tool is the primary entry point for installing and configuring the Skupper infrastructure. You need to install the skupper command only once for each development environment.

*curl https://skupper.io/install.sh | sh*

The script installs the command under your home directory. It prompts you to add the command to your path if necessary.

**Step 2: Configure access to multiple namespaces**

A single kubeconfig supports only one active context per user. Since you will be using *two* contexts at once in this exercise, you need to create two distinct kubeconfigs. You can then use the first kubeconfig in one console session, and the second kubeconfig in another.

* Configure separate console sessions:

Start a console session for each of your namespaces. Set the KUBECONFIG environment variable to a different path in each session.

CONSOLE FOR WEST

*export KUBECONFIG=$HOME/.kube/config-west*

CONSOLE FOR EAST

*export KUBECONFIG=$HOME/.kube/config-east*

* Configure cluster access
* Set the current namespaces

Use kubectl create namespace to create the namespaces you wish to use. Use kubectl config set-context to set the current namespace for each session.

CONSOLE FOR WEST

*kubectl create namespace west*

*kubectl config set-context --current --namespace west*

CONSOLE FOR EAST

*kubectl create namespace east*

*kubectl config set-context --current --namespace east*

**Step 3: Install the Skupper router in each namespace**

The skupper init command installs the Skupper router in the current namespace. start minikube tunnel before you install Skupper.

* Install the router

Run the skupper init command in the West and East namespace.

WEST

*$ skupper init*

EAST

*$ skupper init --ingress none*

**Step 4: Link your namespaces**

After installation, you have the infrastructure you need, but your namespaces are not linked. Creating a link requires use of two skupper commands in conjunction, skupper token create and skupper link create.

* Generate a link token

In West, use the skupper token create command to generate a token.

WEST

*skupper token create $HOME/secret.yaml*

* Use the token to create a link

With the token in hand, you are ready to link the namespaces. Pass the token from West to the skupper link create command in East.

EAST

*skupper link create $HOME/secret.yaml*

**Step 5: Expose your services**

* Deploy the frontend and backend services

Use kubectl create deployment to start the frontend in West.

WEST

*Kubectl create deployment hello-world-frontend --image quay.io/skupper/hello-world-frontend*

EAST

*kubectl create deployment**hello-world-backend --image quay.io/skupper/hello*

* Expose the backend service

Use the skupper expose command in East to make hello-world-backend available in West.

EAST

*skupper expose deployment/hello-world-backend --port 8080*

* Check the backend service

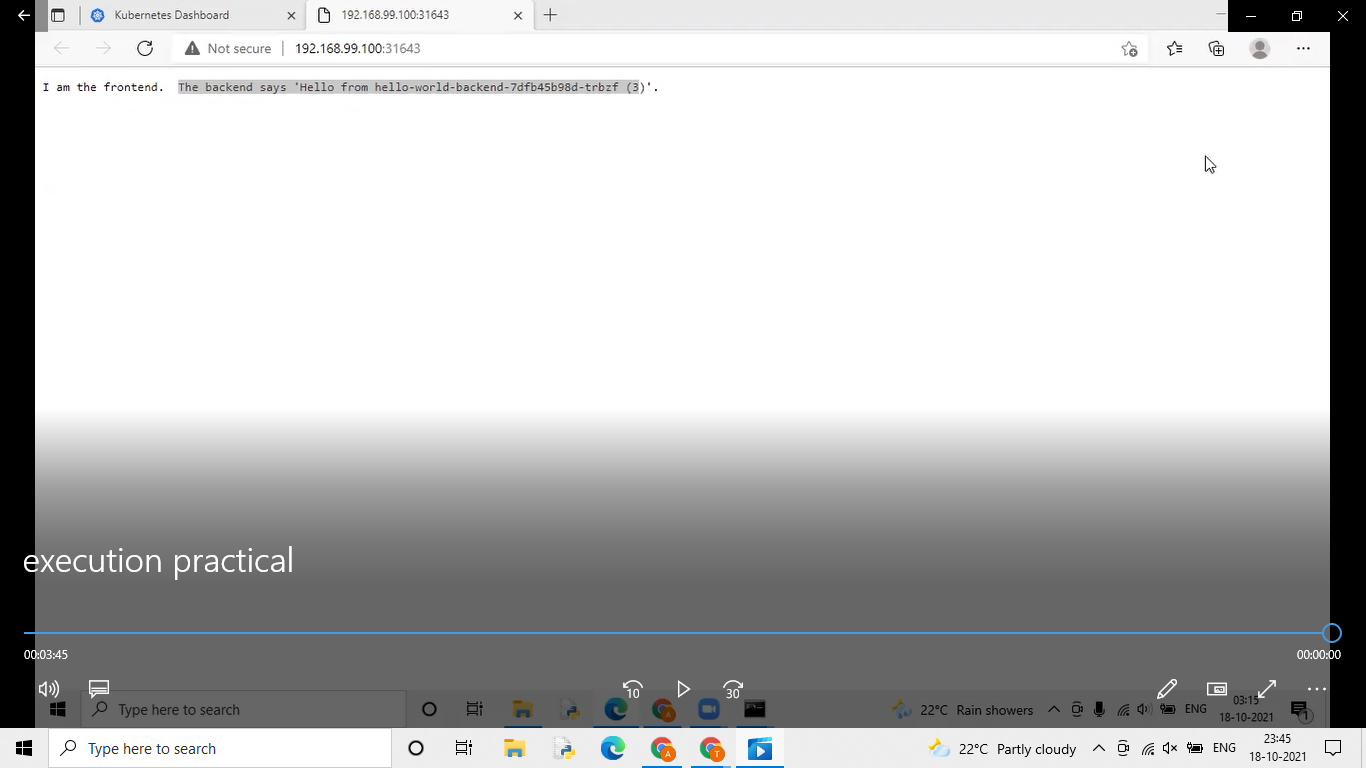
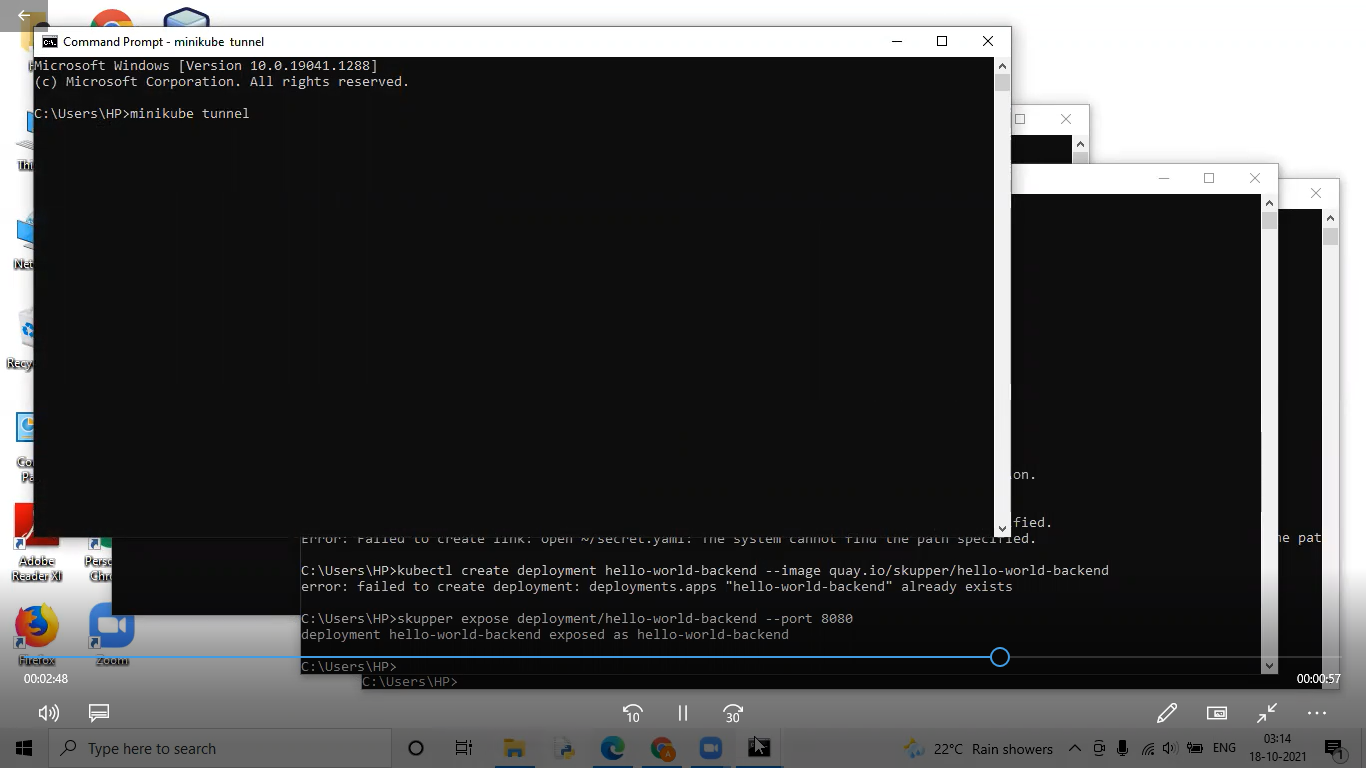
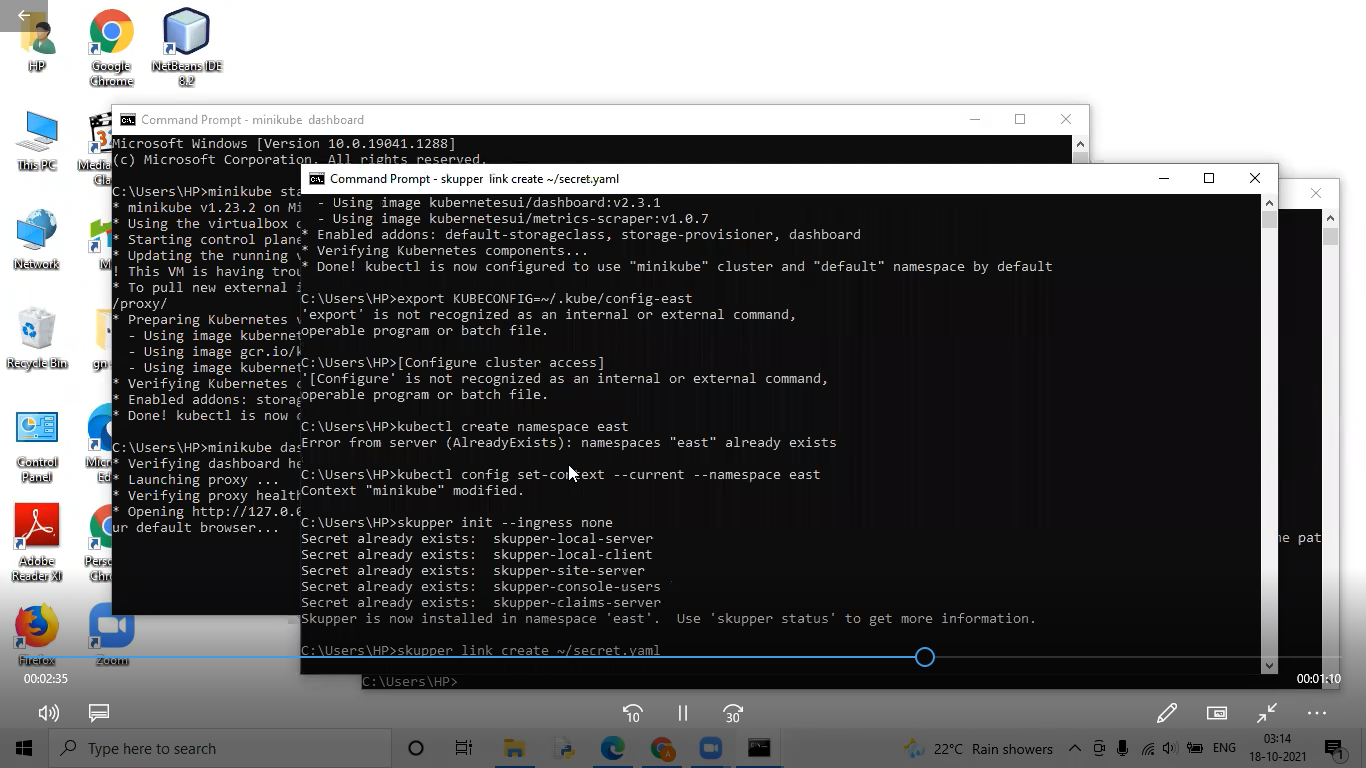
Use kubectl get services in West to make sure the hello-world-backend service from East is represented.

WEST

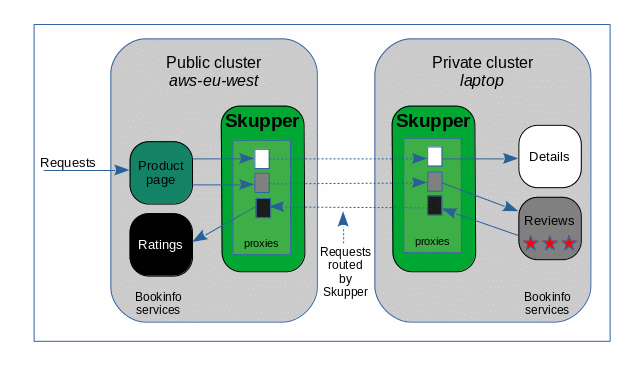
*$ kubectl get services*

*NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE*

*hello-world-backend ClusterIP 10.96.175.18 <none>*

**

BOOKINFO:

****

**Step 1: Deploy the Bookinfo application**

This step creates a service and a deployment for each of the four Bookinfo microservices.

Namespace aws-eu-west:

*$ kubectl apply -f public-cloud.yaml*

*service/productpage created*

*deployment.extensions/productpage-v1 created*

*service/ratings created*

*deployment.extensions/ratings-v1 created*

Namespace laptop:

*$ kubectl apply -f private-cloud.yaml*

*service/details created*

*deployment.extensions/details-v1 created*

*service/reviews created*

*deployment.extensions/reviews-v3 created*

**Step 2: Expose the public productpage service**

Namespace aws-eu-west:

*$ kubectl expose deployment/productpage-v1 --port 9080 --type LoadBalancer*

The Bookinfo application is accessed from the public internet through this ingress port to the productpage service.

create a route:

*$ oc create route edge productpage --service=productpage*

**Step 3: Observe that the application does not work**

The web address for the Bookinfo application can be discovered from namespace aws-eu-west:

*$ echo $(kubectl get service/productpage -o jsonpath='http://{.status.loadBalancer.ingress[0].ip}:9080')*

Open the address in a web browser.

**Step 4: Set up Skupper**

This step initializes the Skupper environment on each cluster.

Namespace laptop:

*$ skupper init*

Namespace aws-eu-west:

*$ skupper init*

Now the Skupper infrastructure is running. Use skupper status in each console terminal to see that Skupper is available.

*$ skupper status*

Skupper is enabled for namespace '"<ns>" in interior mode'. It is not connected to any other sites. It has no exposed services.

**Step 5: Connect your Skupper installations**

Connect your namespaces with a Skupper connection.

The skupper token create <file> command directs Skupper to generate a secret token file with certificates that grant permission to other Skupper instances to connect to this Skupper's network.

The skupper link create <file> command directs Skupper to connect to another Skupper's network. This step completes the Skupper connection.

Generate a Skupper network connection token

Namespace aws-eu-west:

*$ skupper token create ${HOME}/PVT-to-PUB-connection-token.yaml*

Open a Skupper connection

Namespace laptop:

*$ skupper link create ${HOME}/PVT-to-PUB-connection-token.yaml*

Check the connection

Namespace aws-eu-west:

*$ skupper status*

Skupper is enabled for namespace '"aws-eu-west" in interior mode'. It is connected to 1 other site. It has no exposed service.

Namespace laptop:

*$ skupper status*

Skupper is enabled for namespace '"laptop" in interior mode'. It is connected to 1 other site. It has no exposed service.

**Step 6: Virtualize the services you want shared**

This step uses the kubectl annotate command to notify Skupper that a service is to be included in the Skupper network.

Skupper uses the annotation as the indication that a service must be virtualized. The service that receives the annotation is the physical target for network requests and the proxies that Skupper deploys in the other namespaces are the virtual targets for network requests. The Skupper infrastructure then routes requests between the virtual services and the target service.

Namespace aws-eu-west:

*$ kubectl annotate service ratings skupper.io/proxy=http*

*service/ratings annotated*

Namespace laptop:

*$ kubectl annotate service details skupper.io/proxy=http*

*service/details annotated*

*$ kubectl annotate service reviews skupper.io/proxy=http*

*service/reviews annotated*

**Step 7: Observe that the application works**

The web address for the Bookinfo app can be discovered from namespace aws-eu-west:

*$ echo $(kubectl get service/productpage -o jsonpath='http://{.status.loadBalancer.ingress[0].ip}:9080')*

Open the address in a web browser. The application now works with no errors.

