**Demo Part**

**Guestbook App with Redis**

The guestbook application is a multi-tier web application that uses Redis and Nginx, with Kubernetes.

We have used Typescript for coding.

**Objectives**

* Start-up Redis leader
* Start-up Redis replicas
* Start up the guestbook frontend
* Expose and view the frontend service
* Clean up

**Prequisites**

* Install Pulumi locally
* Install NPM locally
* Install IBM Cloud CLI locally
* Created a Kubernetes cluster on IBM Cloud Platform
* Download kubectl to verify cluster is and up and Running
* Connect Pulumi to a Kubernetes Cluster

**Create and Configure a Project**

* To create a new Pulumi Project

Commands: mkdir k8s-guestbook

&& cd k8s-guestbook && pulumi new kubernetes-typescript

* Modify the index.ts file for your convenience
* Set Load balancer as true to expose frontend service

**Connecting Kubernetes Cluster to IBM**

* Check whether Pulumi CLI and Kubectl is installed
* Check for Kubeconfig file under .kube/config
* To connect to Kubernetes Cluster on IBM cloud perform below steps
* kubectl config set-context my-context --cluster=my-cluster --user=my-user ( Cluster name and user to be got from Config file)
* pulumi stack init new-kube-stack && pulumi config set kubernetes:context my-context
* kubectl config use-context my-context

**Deploying**

Use the Pulumi up command to execute the code

**Command** : pulumi up

**Viewing the Guestbook**

The application is now running in our cluster. Let’s inspect our cluster state to validate the deployment

**Command**: kubectl get services

**No Load Balancer**

To access the frontend service locally use the port forward command :

kubectl port-forward svc/frontend 8765:80

**To clean up the deployment:**

Clean up and destroy the resources and remove our stack:

**Command:** pulumi destroy --yes && pulumi stack rm --yes

**To expose using cluster public ip:**

ibmcloud ks worker ls --cluster <cluster\_name\_or\_ID>