Kubernetes Assignment Solution

1. Creating a Kubernetes Cluster using Minikube

- Minikube allows us to set up a single-node Kubernetes cluster locally for learning and testing purposes.
- Steps:
- - Installed Docker, Minikube, and kubectl.
- - Started the cluster using: minikube start
- - Verified with: kubectl cluster-info and minikube status
- - Accessed dashboard using: minikube dashboard
- Outcome: A local Kubernetes cluster was successfully created.

2. Creating a Kubernetes Cluster using kubeadm

- Kubeadm provides a manual method for creating multi-node Kubernetes clusters.
- Steps:
- - Prepared 2 Ubuntu machines (Master + Worker).
- - Installed kubelet, kubeadm, and kubectl.
- - Initialized master using: sudo kubeadm init
- - Configured kubectl and joined the worker using kubeadm join.
- Outcome: Multi-node cluster setup successfully.

3. Deploying AKS Cluster via Azure Portal

- Steps:
- - Logged into Azure Portal and created AKS service.
- - Chose VM size, enabled monitoring, and deployed.
- - Used Azure CLI to access: az aks get-credentials
- - Configured roles with Role and RoleBinding YAMLs.
- Outcome: AKS cluster deployed and accessed with user roles configured.

4. Deploying Microservice App & Public Access

- Steps:
- - Created deployment and service YAMLs for NGINX app.
- - Used LoadBalancer type service to expose app publicly.

- - Retrieved public IP with: kubectl get svc
- Outcome: App deployed and accessed via public internet.

5. Exposing Services (ClusterIP, NodePort, LoadBalancer)

- ClusterIP: Internal service within cluster.
- NodePort: Exposed on a specific port on each node.
- LoadBalancer: Used in cloud to expose app to internet.
- Outcome: All three service types tested successfully.

6. YAML Files Summary

- nginx-deployment.yaml
- nginx-clusterip-service.yaml
- nginx-nodeport-service.yaml
- nginx-loadbalancer-service.yaml
- rbac-user-role.yaml