1. Write a common use-case, where you will use a daemon set instead of replica set.

To collect metrics and logs from every node in your Kubernetes cluster for monitoring and troubleshooting purposes.

A DaemonSet is ideal for this because one instance per node, automatic scaling and long-running process.

2. Suppose you have deployed your application using a deployment controller. Assume the initial number of replicas is one. Write the steps needed to update a container's image using deployment, making sure that there is zero downtime.

\*Update Deployment Configuration:

Edit the deployment YAML file to specify the new container image.

\*Apply the Updated Configuration:

Use kubectl apply -f <deployment-file>.yaml to apply the changes.

\*Verify Rolling Update:

Monitor the update using kubectl rollout status deployment/<deployment-name>.

\*Check Pod Status:

Ensure all new pods are running with kubectl get pods.

\*Verify Zero Downtime:

Test the application to confirm it remained accessible during the update.

3. You have deployed an application, that is listening at port 8000. You used a replica-set to deploy it and created a NodePort service to make it accessible. But when you test it, somehow the application is not reachable at the port. Write down your approach and sequentially all the steps that you will take to find out the issue.

\*Check Pod Status:

Use kubectl get pods to ensure all pods are running.

\*Verify ReplicaSet Status:

Run kubectl get rs to confirm the ReplicaSet is managing the desired number of pods.

\*Inspect Pod Logs:

Use kubectl logs <pod-name> to check for any errors in the application logs.

\*Verify Service Configuration:

Run kubectl get svc <service-name> to check if the NodePort service is correctly configured and listening on the expected port.

\*Describe Service:

Use kubectl describe svc <service-name> to get detailed information and check for any misconfigurations.

\*Check NodePort Accessibility:

Ensure the NodePort is within the valid range (30000-32767) and is correctly exposed on the nodes.

\*Test Port Reachability:

From a node, use curl http://<node-ip>:<node-port> to verify the application is accessible via the NodePort.

\*Check Firewall Rules:

Ensure no firewall rules are blocking access to the NodePort on the nodes.

\*Network Policies:

Verify there are no network policies that might be restricting access to the service.

\*Recreate NodePort Service:

If everything seems correct but the issue persists, delete and recreate the NodePort service to ensure there are no hidden configuration issues.