# Kubernetes Lab Report

## 1. Orchestration tools in modern applications

(a) Orchestration tools like Kubernetes help manage and scale application servers by automating container scheduling, scaling, networking, and load balancing. They ensure applications remain available even when some nodes fail.  
  
(b) These tools automate deployment, scaling, and management of applications through declarative configuration files. Kubernetes automatically handles container placement, scales replicas based on resource usage, and performs self-healing when containers crash.

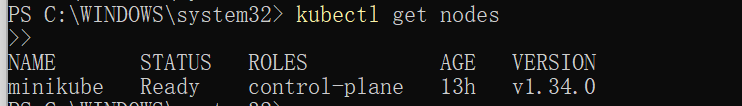
## 2. Difference between Pod, Deployment, and Service

• Pod: The smallest deployable unit in Kubernetes, which encapsulates one or more containers.  
• Deployment: Manages a set of Pods, ensuring the desired number of replicas are running and enabling rolling updates.  
• Service: Provides a stable network endpoint (IP/DNS) to access a group of Pods, even when Pods are recreated.

## 3. Namespace

A Namespace provides a logical partition within a Kubernetes cluster for separating resources and users.  
Example: default, kube-system, kube-public.

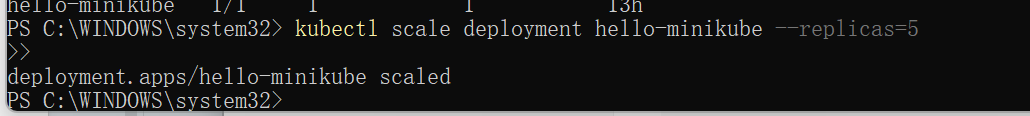
## 4. Role of the Kubelet

The Kubelet runs on each node and ensures that containers described in PodSpecs are running and healthy.  
Command to check nodes:  
kubectl get nodes  
  


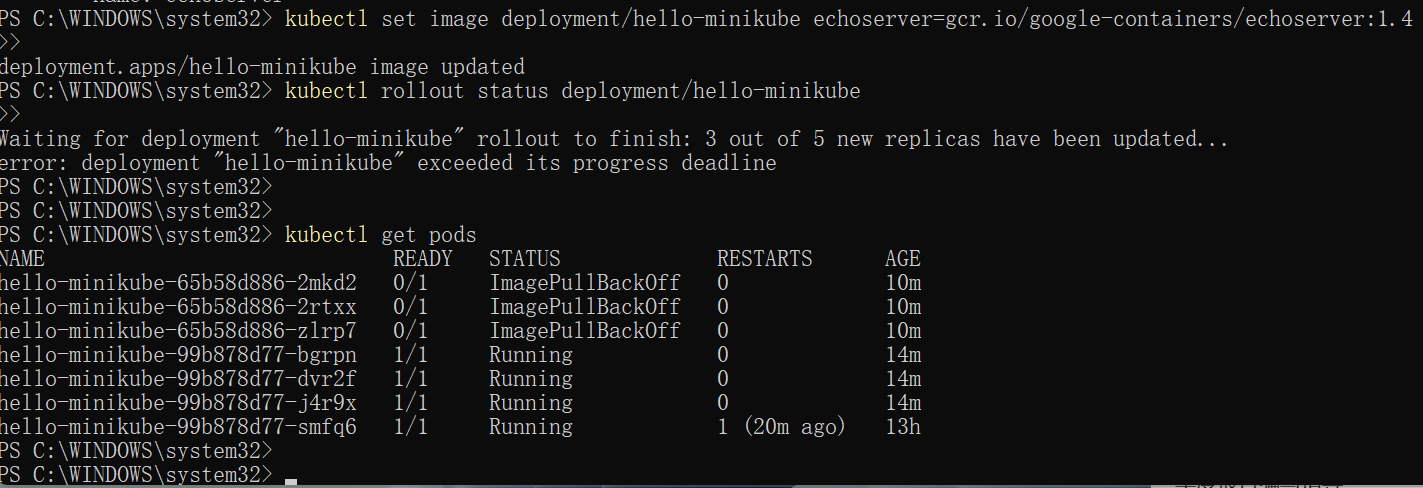
## 5. ClusterIP vs NodePort vs LoadBalancer

ClusterIP: Internal-only communication.  
NodePort: Exposes the service on each node’s IP and static port.  
LoadBalancer: Exposes externally via cloud provider load balancer.  
  
[Insert Screenshot Here]

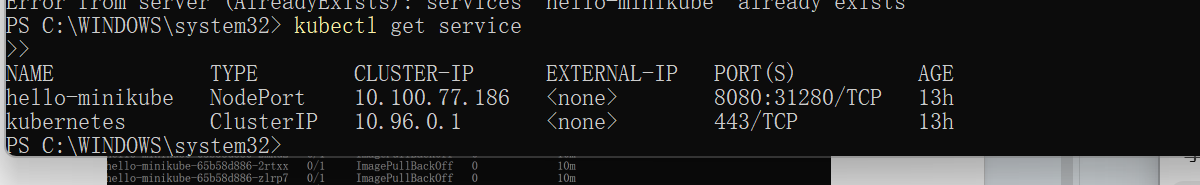
## 6. Scaling a Deployment to 5 replicas

Command:  
kubectl scale deployment <deployment-name> --replicas=5  


## 7. Updating a Deployment image without downtime

Command:  
kubectl set image deployment/<deployment-name> <container-name>=nginx:latest  
  
Kubernetes performs a rolling update, gradually replacing old Pods with new ones.  
  


## 8. Exposing a Deployment to external traffic

Command:  
kubectl expose deployment <deployment-name> --type=NodePort --port=8080  
  


## 9. Kubernetes Scheduling

Kubernetes uses a scheduler to assign Pods to nodes based on resource availability, taints/tolerations, affinities, and priorities.

## 10. Role of Ingress

### Ingress manages external HTTP/HTTPS access to Services using routing rules. It provides URL-based routing and SSL termination, unlike Services that expose ports.

### **Experiment Conclusion**

In this experiment, I successfully explored the fundamental concepts and practical operations of Kubernetes using Minikube as a local cluster environment. Through a series of steps — including deploying an application, exposing it to external traffic, scaling replicas, and updating container images — I gained a comprehensive understanding of container orchestration.

I learned how Kubernetes automates deployment, scaling, and management of containerized applications through its key components such as **Pods, Deployments, and Services**. Additionally, I practiced using essential kubectl commands to monitor resources, modify configurations, and ensure high availability during rolling updates.

During the process, I also encountered and resolved several common issues, such as image pulling errors, port conflicts, and service duplication, which deepened my troubleshooting experience. Overall, this lab provided hands-on experience in container orchestration, enhanced my understanding of DevOps workflows, and improved my confidence in managing cloud-native applications.