📘 Kubernetes Notes

🧠 1. What is Kubernetes?

- Kubernetes is an **open-source container orchestration system** to automate deployment, scaling, and management of containerized applications.
- It ensures high availability, load balancing, and fault tolerance for containers across multiple machines.

🏛️ 2. Kubernetes Architecture

Kubernetes has two main categories of components:

📍 Control Plane (Master Node)

Responsible for overall cluster management.

- **API Server:** Exposes Kubernetes API, central point for management.
- **Scheduler:** Places Pods on worker nodes based on constraints.
- **Controller Manager:** Runs controllers (Node Controller, Replication Controller, etc.).
- **etcd:** Key-value store storing cluster state.

📍 Worker Nodes

Nodes run application workloads.

- **Kubelet:** Ensures containers described in specs are running.
- **Kube-Proxy:** Maintains networking rules for pods & services.
- **Pods:** Smallest deployable unit in Kubernetes consisting of containers.

🔄 3. Pod Lifecycle

1. Pod manifest is submitted to the API Server.
2. Scheduler places it on a suitable node.
3. Node pulls container images and starts containers.
4. Pod moves to **Running** state if successful.

5. On completion or failure, the Pod changes state accordingly.

📄 4. Kubernetes Manifest (Pod Example)

A **manifest file** defines Kubernetes objects (usually YAML).
Example **Pod manifest**:

```
apiVersion: v1

kind: Pod

metadata:

  name: nginx-pod

  labels:

    app: nginx

    tier: dev

spec:

  containers:

    - name: nginx-container

      image: nginx
```

- **apiVersion:** Version of Kubernetes API (e.g., v1).
- **kind:** Type of Kubernetes object.
- **metadata:** Name, namespace, labels.
- **spec:** Desired state of the object.

🚀 5. How to Apply the Manifest

To create the Pod:

```
kubectl create -f nginx-pod.yaml
```

Check Pods:

```
kubectl get pods
```

✓ You should see the Pod running.

🛠️ Kubernetes Installation Overview (Ways to Get Started)

You can set up Kubernetes in various ways depending on your environment and goals. Kubernetes

---

🔧 1. Play-with-k8s (Online Playground)

- Quick testing environment hosted online.
- Good when you don't want to install Kubernetes locally.

💡 Use this for experimentation without setup hassle.

🏠 2. Minikube (Local Learning Cluster)

- Great for Windows, macOS, and Linux learning setups.
- Runs a **single-node cluster** in a VM or Docker container.

🧠 Ideal for beginners to practice kubectl and workloads.

🧩 3. kind (Kubernetes in Docker) *(not in original blog but relevant)*

- Runs Kubernetes clusters **inside Docker containers**.
- Lightweight and great for local development/testing.

💡 Works well on WSL2 and Docker Desktop environments.

🛠️ 4. kubeadm (Production-style Setup)

- Used to set up **real multi-node Kubernetes clusters**.
- Requires multiple VMs/servers.

Ideal for learning advanced cluster management.

☁️ 5. Cloud Managed Kubernetes

- AWS EKS, GCP GKE, Azure AKS, etc.
- Cloud handles control plane and infrastructure.

💡 Best for production workloads.

🧪 kubectl — Kubernetes CLI

- kubectl is the command-line tool to communicate with the Kubernetes API.
- You use it to create, inspect, update, and delete Kubernetes objects.

Examples:

kubectl get pods

kubectl describe pod nginx-pod

kubectl logs <pod-name>

⚠️ Common Installation Mistakes & Things to Avoid

❌ Mistake 1: Installing WSL commands in Linux

- wsl --shutdown only works in **Windows PowerShell**, not inside Linux. Commands like apt install wsl are incorrect.

❌ Mistake 2: Not enabling Docker Desktop WSL integration

If you use WSL2, ensure Docker Desktop is integrated with your distro. Otherwise, tools like Minikube or kind cannot talk to Docker.

❌ Mistake 3: Enabling Docker Desktop Kubernetes and kind together

- Don't enable built-in Kubernetes in Docker Desktop when using kind/Minikube — can lead to conflicts.

❌ Mistake 4: Not checking resource requirements

- Kubernetes needs **sufficient CPU/RAM**.
- On Windows/Mac, ensure virtualization or WSL2 is enabled.

❌ Mistake 5: Running installation tools wrong platform

- kubeadm is for Linux servers — not directly on Windows or macOS.

- For Windows/macOS learning, use Minikube/kind instead. Kubernetes

---

🧠 Quick CLI Commands (Handy Reference)

Purpose	Command
Apply manifest	kubectl apply -f <file>
List Pods	kubectl get pods
Describe object	kubectl describe pod <name>
Delete resource	kubectl delete -f <file>
Get cluster info	kubectl cluster-info