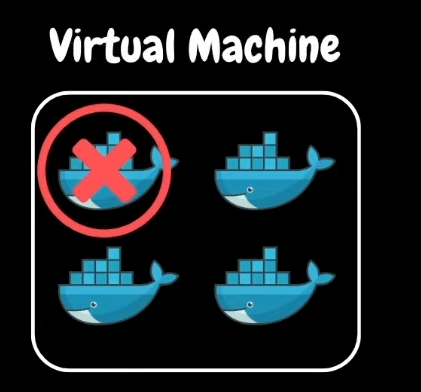
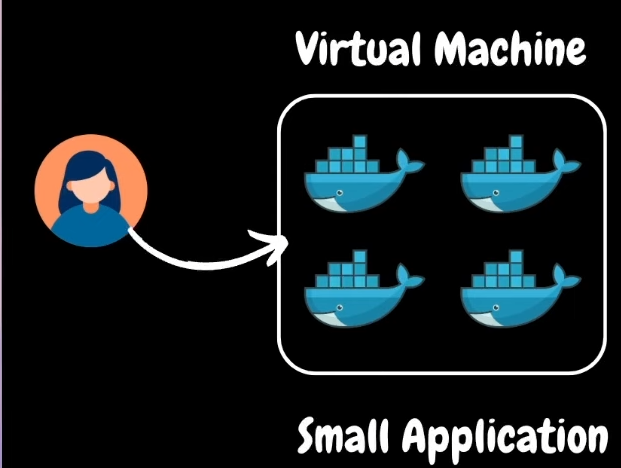
Why Kubernetes needed?

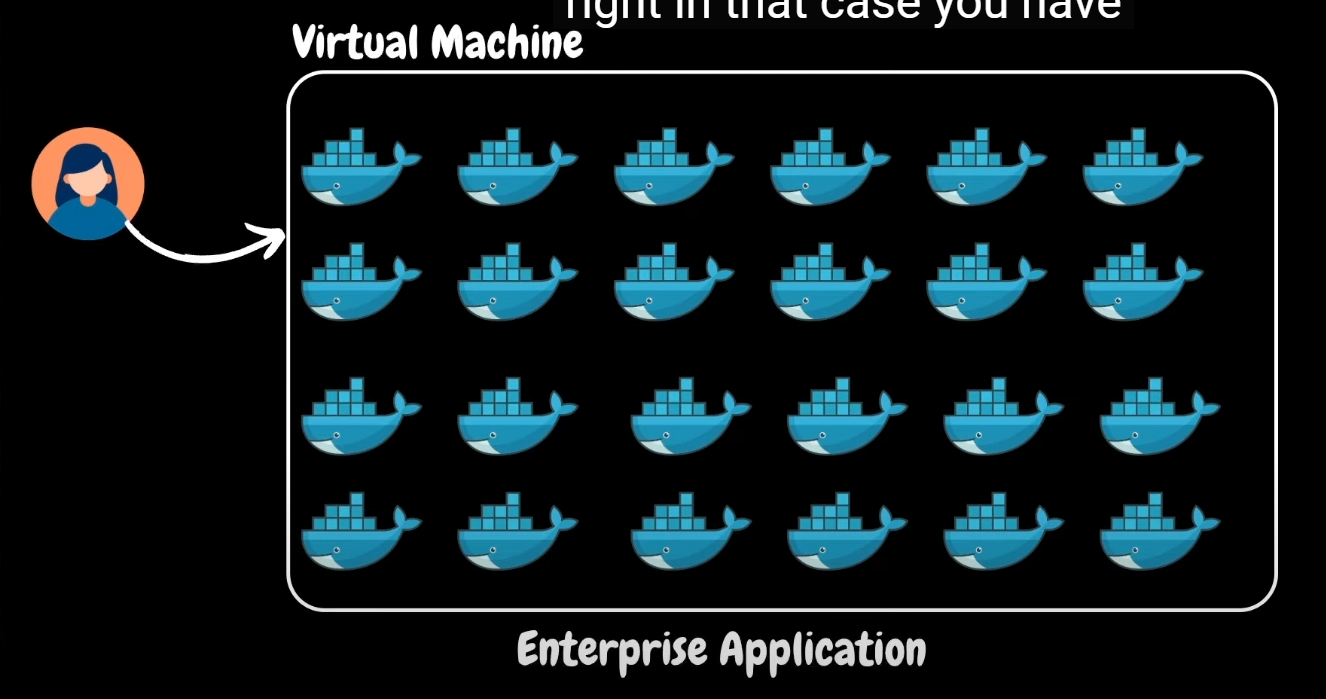


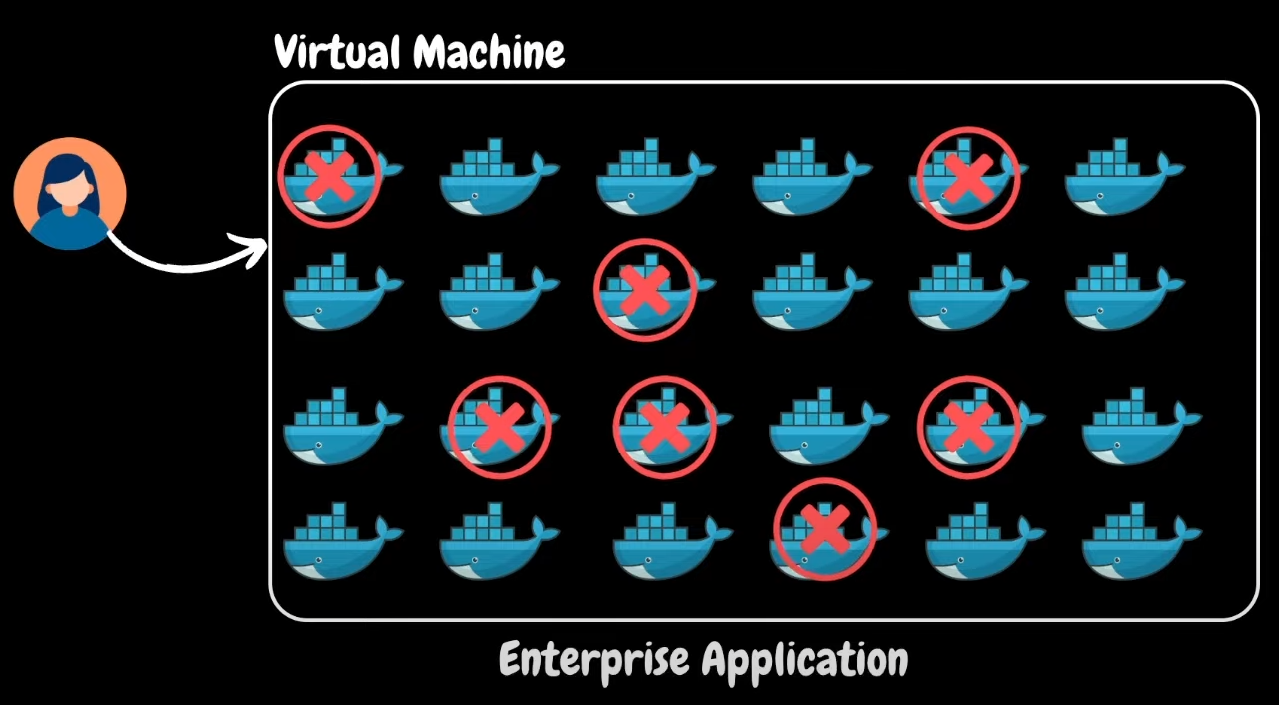
If any of the docker container is getting failed or not working then who takes care of it ?

So for basic any viewer can come and see the what is the problem then it will fix it then again working normally



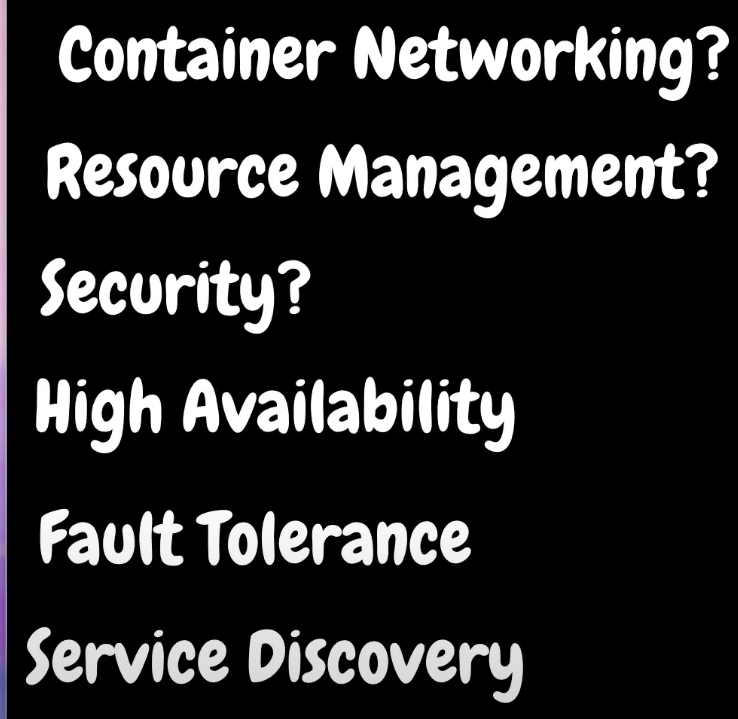
but this is for very small application right . but whenever you are dealing with the big application at that time what? and another thing is that you have to monitor the container whole the time 24/7.





When deploying a new version of an application (e.g., version 1.0) across hundreds of containers, manually managing the updates can be cumbersome. Kubernetes helps automate deployment and manage networking, security, and resource allocation efficiently.

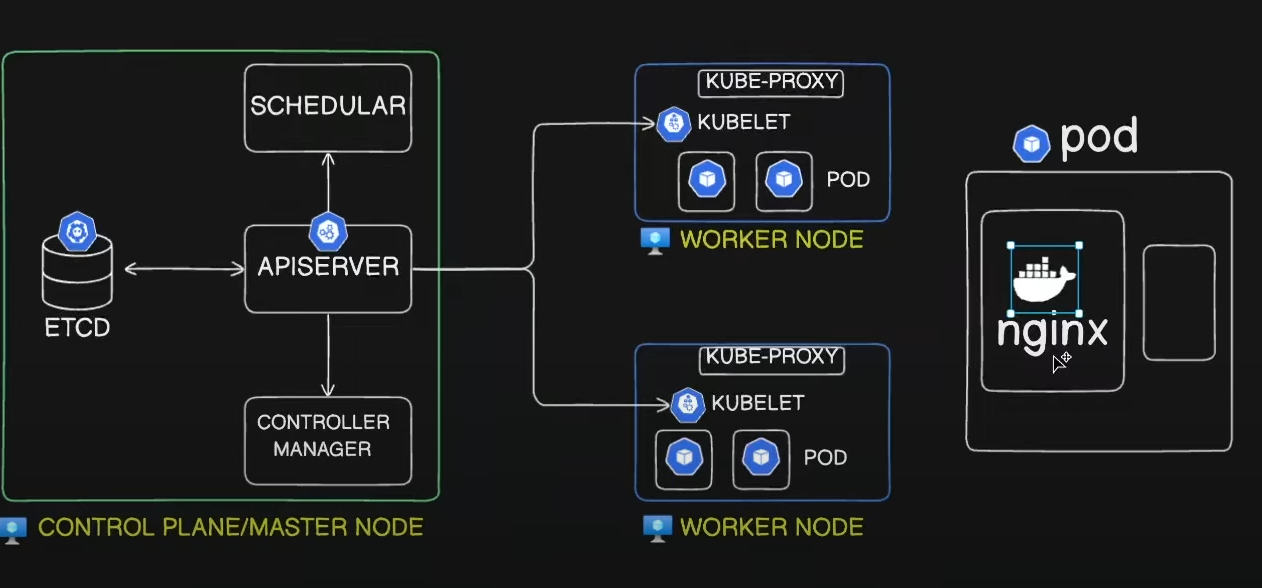
There are lot of challenges



For small applications with only a couple of containers, using an orchestration system like Kubernetes can be inefficient and costly. It leads to unnecessary resource wastage, higher expenses, and increased administrative overhead.

Before deciding on Kubernetes, evaluate if simpler solutions like Docker Compose or running containers on a virtual machine suffice, as they offer lower cost and administrative effort. Kubernetes is essential when scaling and automation are required for larger, more complex applications.

Day 05



**Kubernetes Architecture**

* Overview
  + Control plane (Master node) and worker nodes.
  + Nodes are virtual machines running components and workloads.
* Control Plane (Master Node)
  + Hosts administrative components.
  + Ensures smooth cluster operation.
  + Components: API Server, Scheduler, Controller Manager, etcd.
* Worker Nodes
  + Run actual workloads (containers).
  + Components: Kubelet, Kube-proxy.

**Control Plane Components**

* API Server
  + Central component of the control plane.
  + Handles incoming requests from clients.
  + Interacts with other components.
* Scheduler
  + Schedules workloads (pods) on suitable nodes.
  + Considers constraints like CPU, memory, storage.
* Controller Manager
  + Monitors and manages cluster components.
  + Ensures pods, nodes, and deployments are healthy.
* etcd
  + Key-value data store.
  + Stores cluster state and configuration.
  + Updated by API Server.

**Worker Node Components**

* Kubelet
  + Agent running on each worker node.
  + Receives instructions from the control plane.
  + Manages pods on the node.
* Kube-proxy
  + Enables networking within the node.
  + Creates IP table rules for pod-to-pod communication.

**Example Workflow**

* User Interaction
  + User sends requests via kubectl (CLI tool).
  + API Server authenticates and validates requests.
* Pod Creation
  + API Server updates etcd with pod creation request.
  + Scheduler finds a suitable node.
  + Kubelet on the node creates the pod.
  + API Server updates etcd and sends a response to the user.
* Pod Information Retrieval
  + API Server retrieves pod information from etcd.
  + Sends the response back to the user.

**Conclusion**

* Recap
  + Authenticate and validate requests.
  + Update etcd with cluster state.
  + Scheduler finds nodes for pods.
  + Kubelet manages pods on worker nodes.

What is Pod?

However, kubernetes does not deploy containers directly on the worker nodes. The containers are encapsulated into a Kubernetes object known as PODs. A POD is a single instance of an application. A POD is the smallest object, that you can create in kubernetes.

