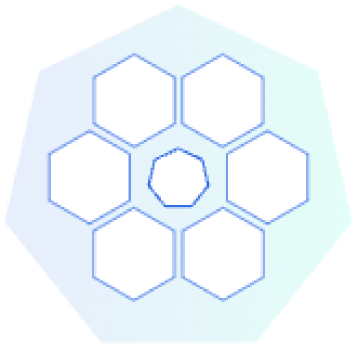
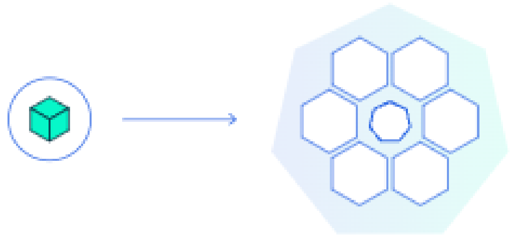
Kubernetes

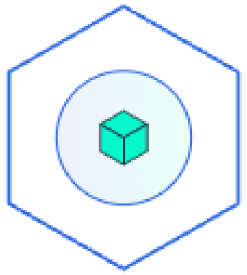
Kubernetes Basics Modules



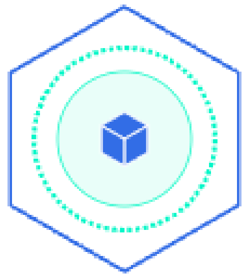
1. Create a Kubernetes cluster



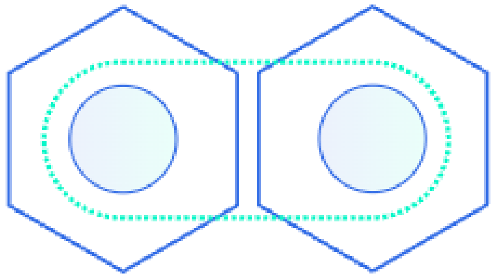
2. Deploy an app



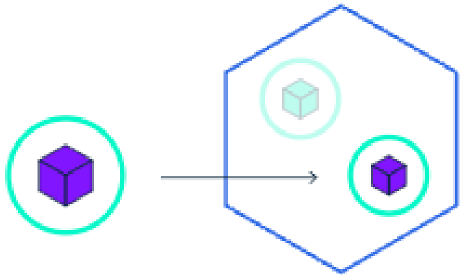
3. Explore your app



4. Expose your app publicly

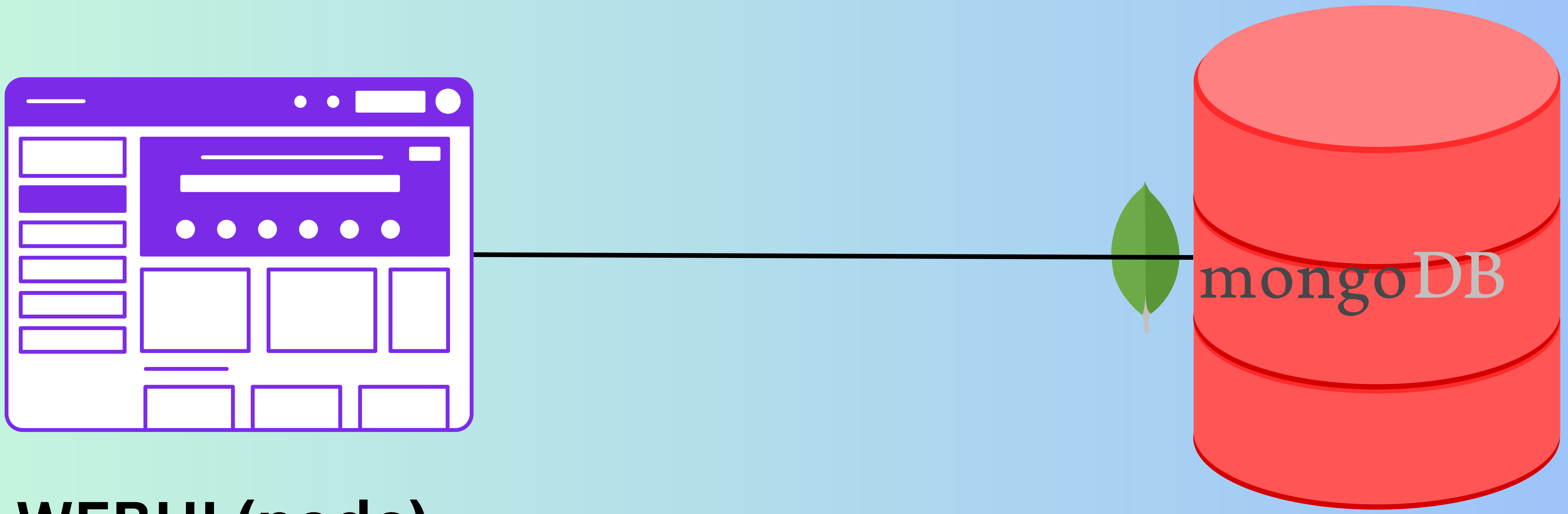


5. Scale up your app



6. Update your app

Multi Container Application



WEBUI (node)

mongodb

Volumes & Data

What is Kubernetes?

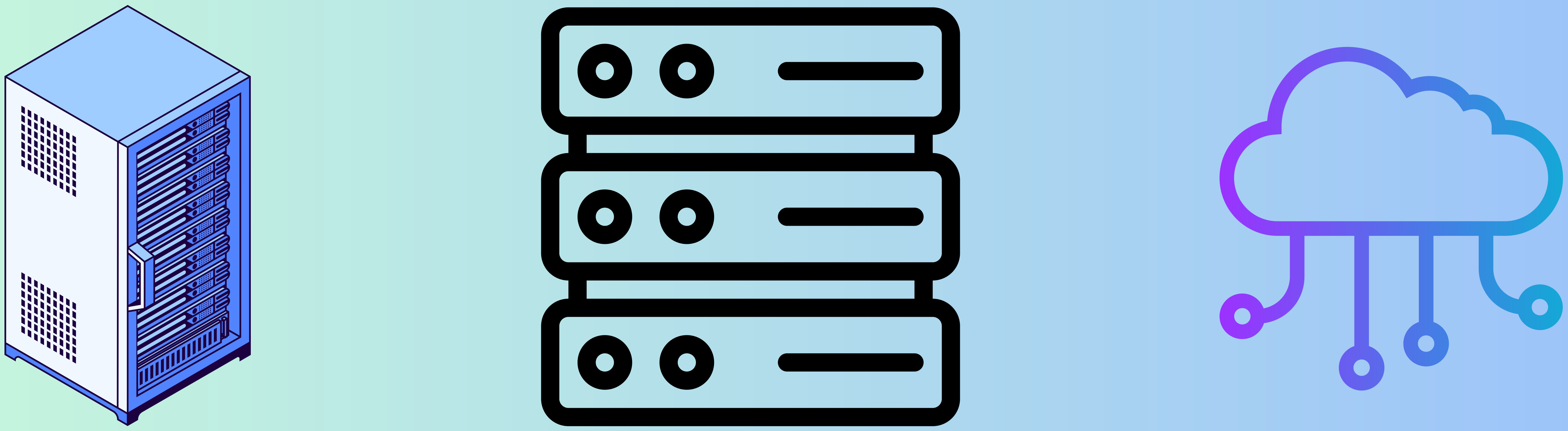
Also known as K8s,
It is an open-source system for -

Automating Deployment

Management of Containerized apps

Scaling

Support different infra..



Developed by - **GOOGLE**

But now maintained by -

Cloud Native Computing Foundation (CNCF)

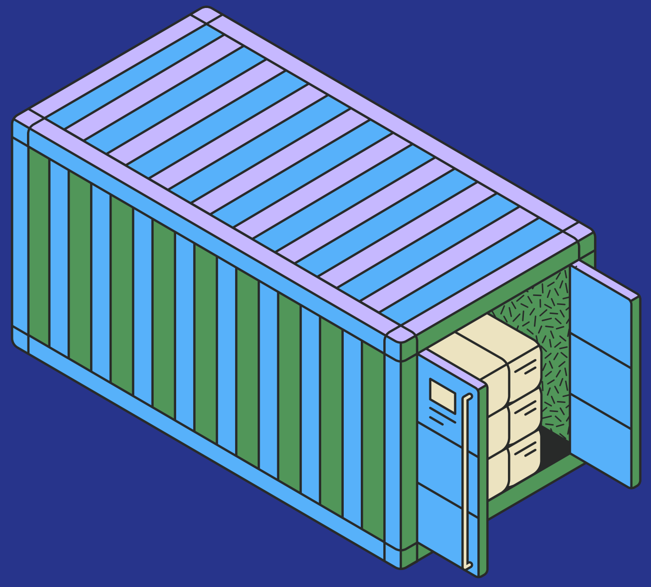
What is Container Orchestration?

Orchestra

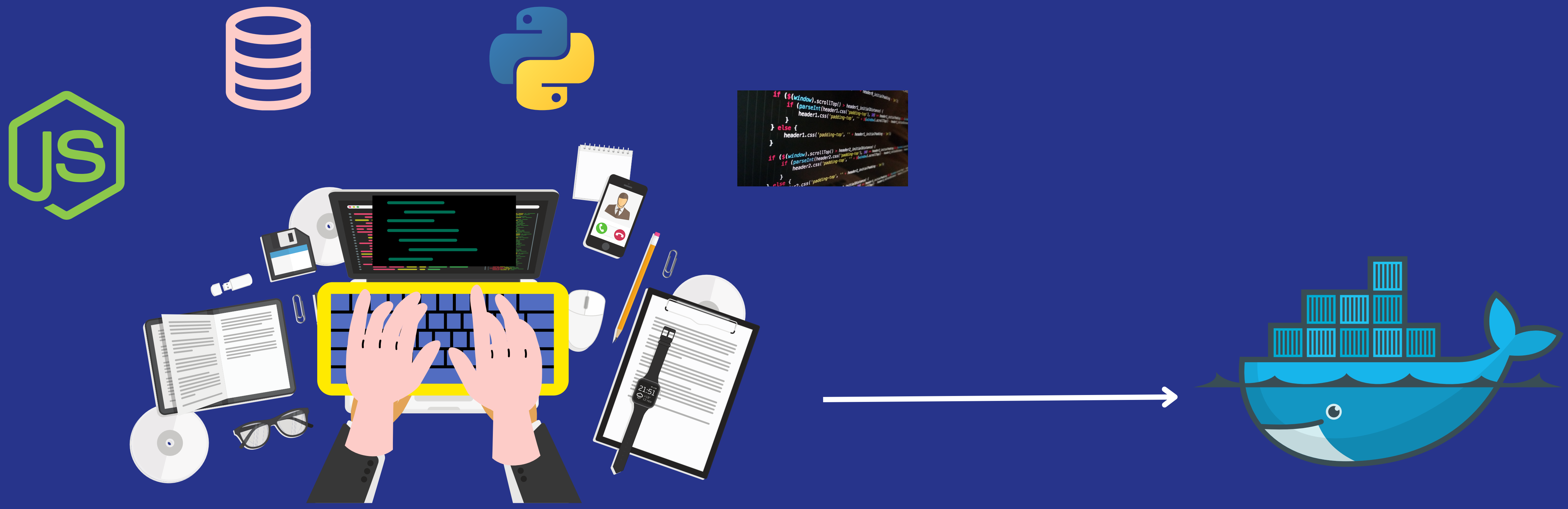
Orchestra

Orchestra

What is a Container?

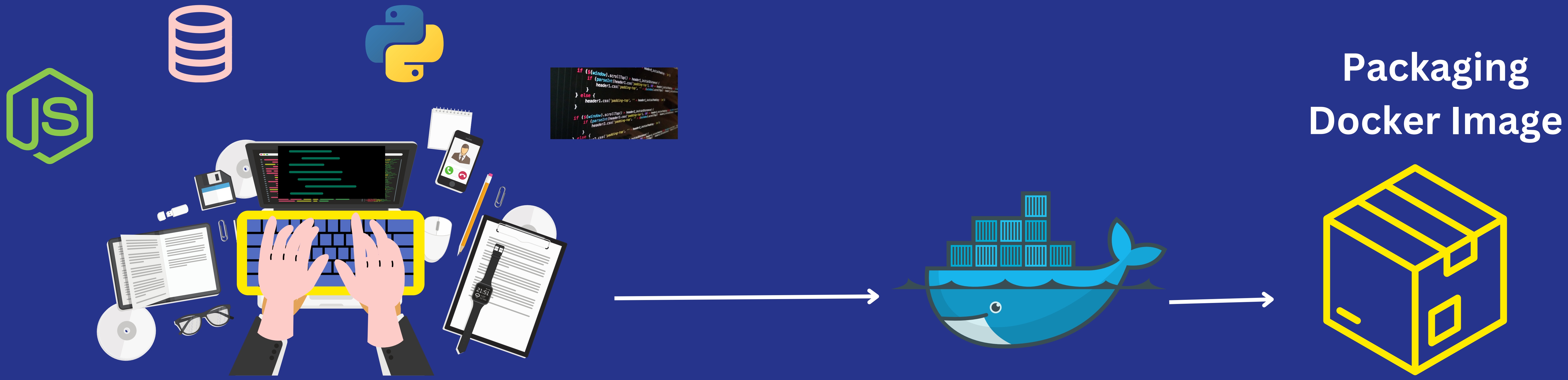


- A way to package an application with all the necessary dependencies and configuration.
- It can be easily shared
- Makes deployment and development efficient.



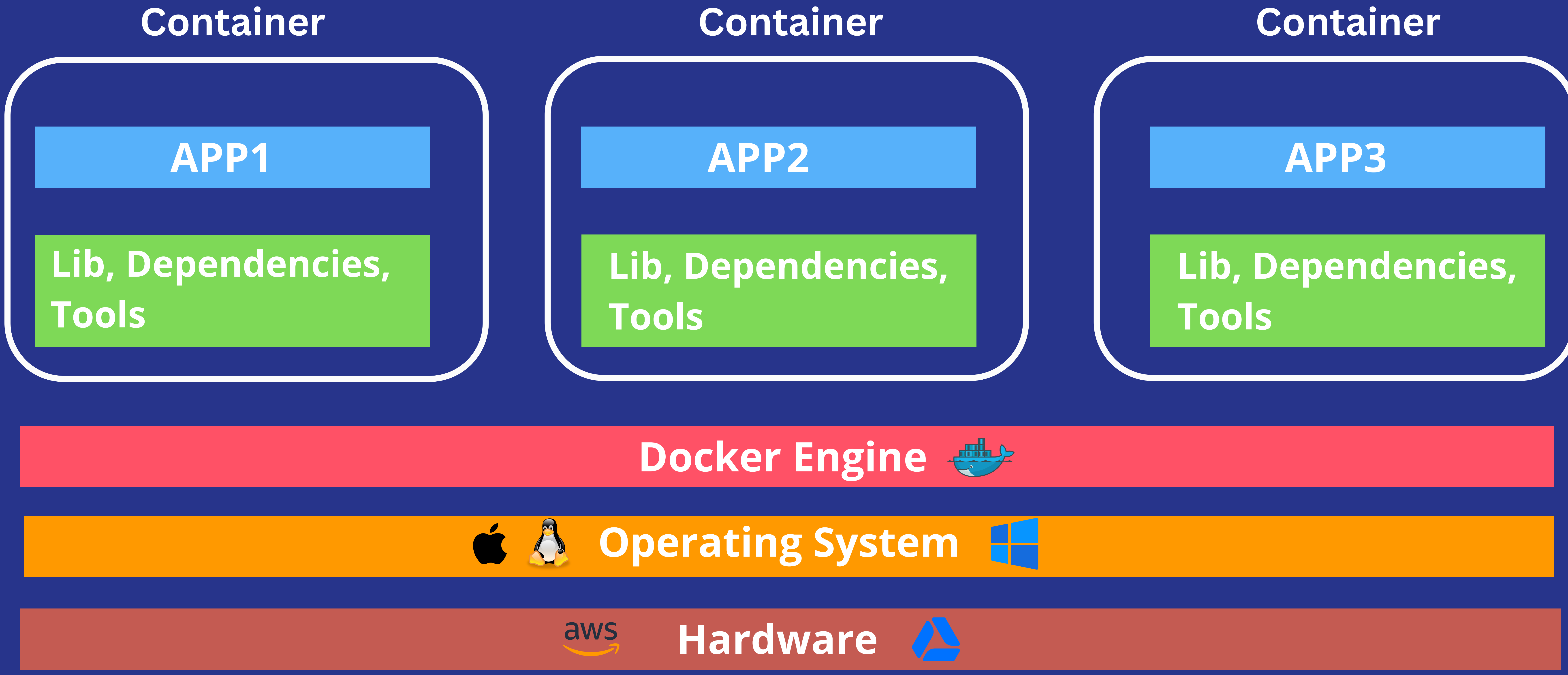
```
if ($(window).scrollTop() > header1_initialDistance) {
    if (parseInt(header1.css('padding-top'), 10)
        header1.css('padding-top', '' + $(window).scrollTop()
    }
} else {
    header1.css('padding-top', '' + header1_initialPadding
}

if ($(window).scrollTop() > header2_initialDistance) {
    if (parseInt(header2.css('padding-top'), 10)
        header2.css('padding-top', '' + $(window).scrollTop()

    }
} else {
```

Developer



Packaging
Docker Image

Developer



Container

APP1

Lib, Dependencies, Tools

Container

APP2

Lib, Dependencies, Tools

Container

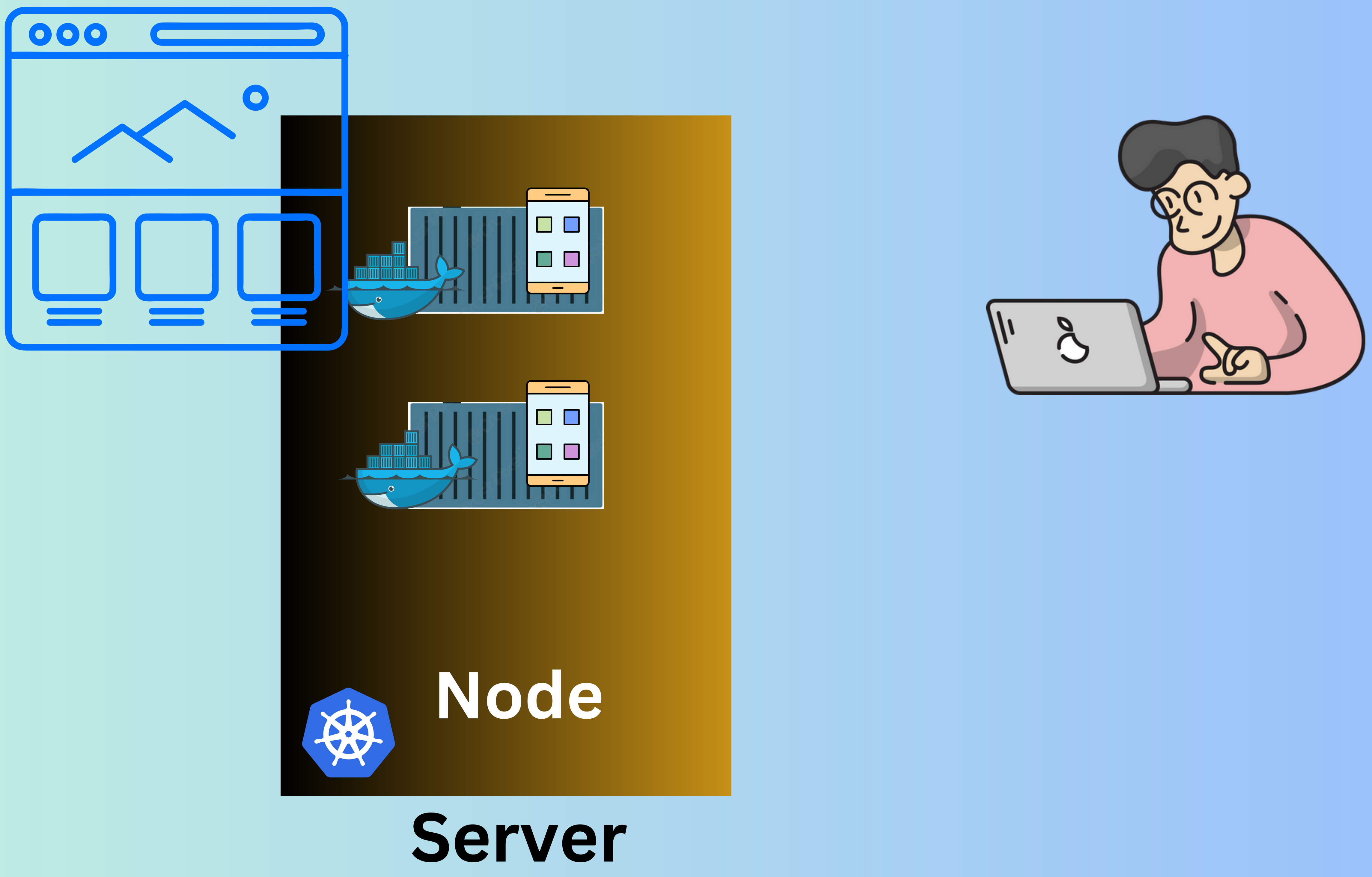
APP3

Lib, Dependencies, Tools

Docker Engine

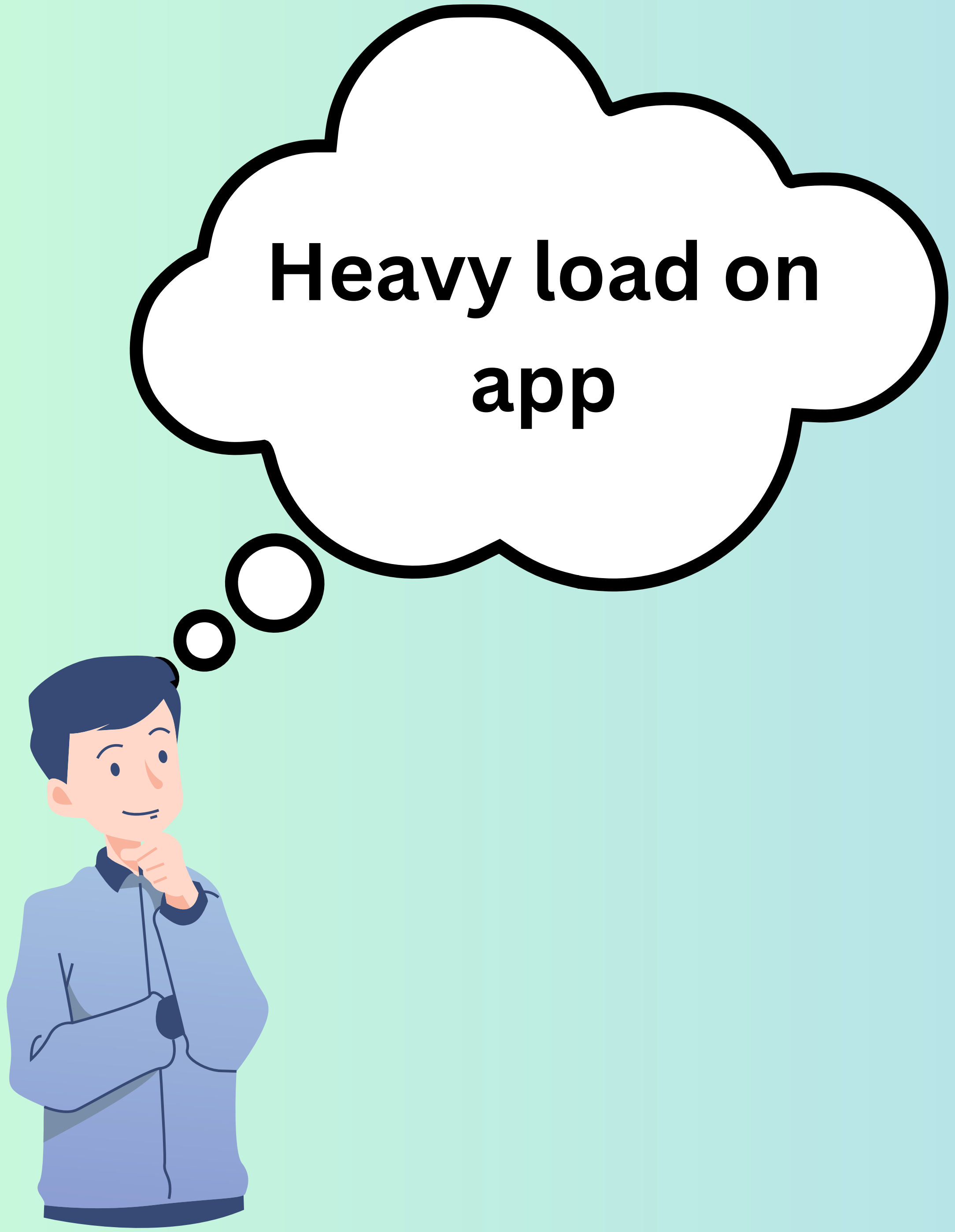
Operating System

Hardware

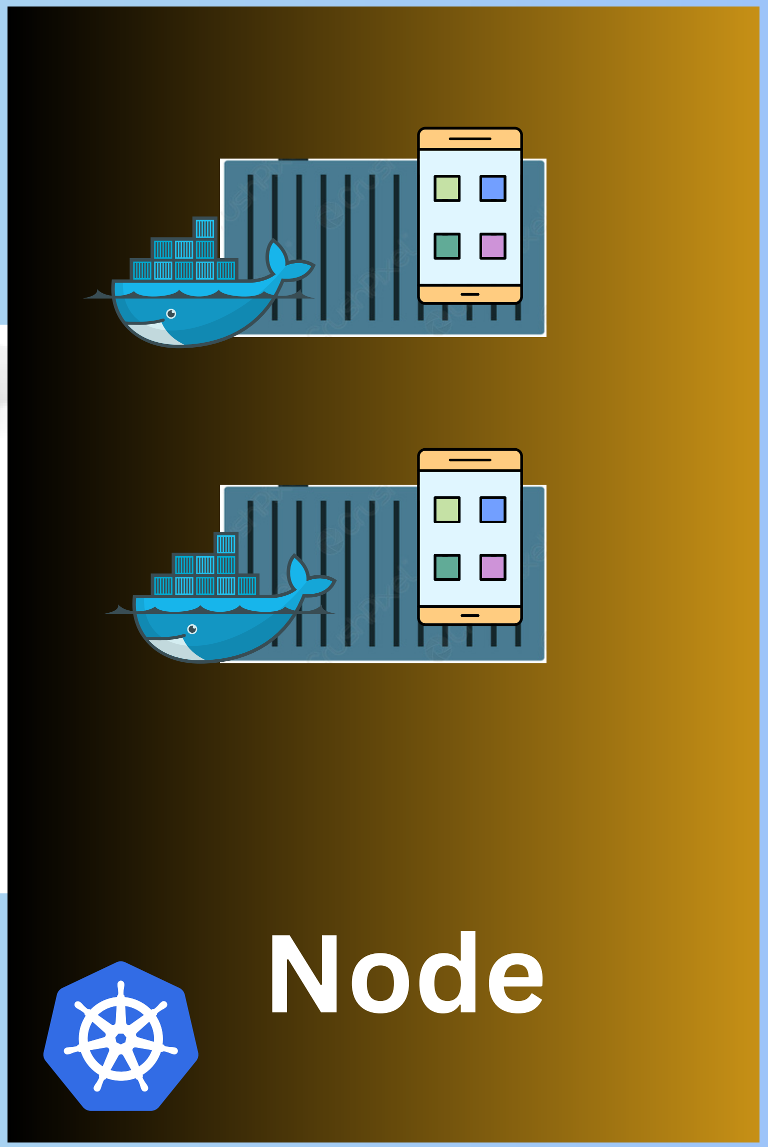


Node

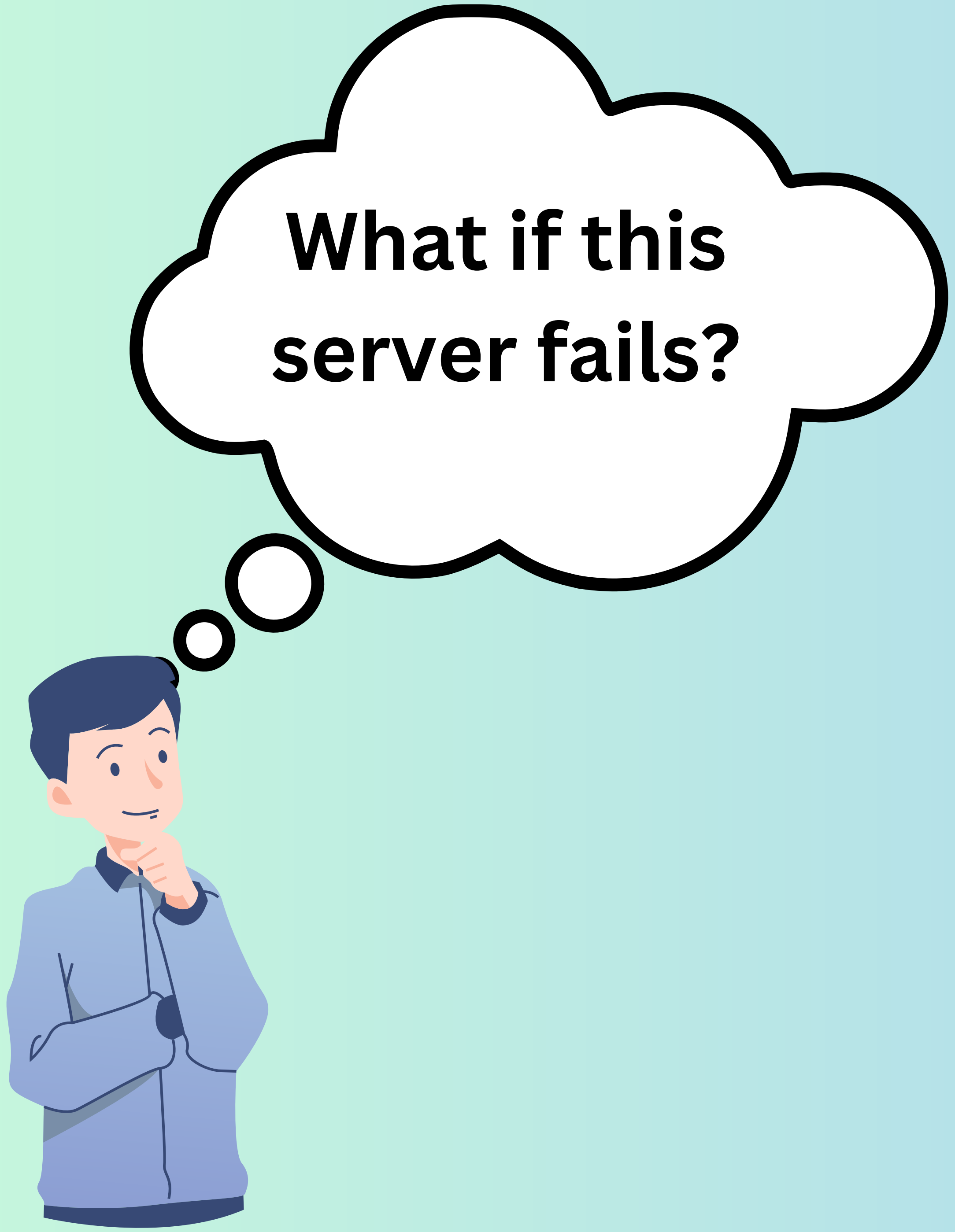
Server

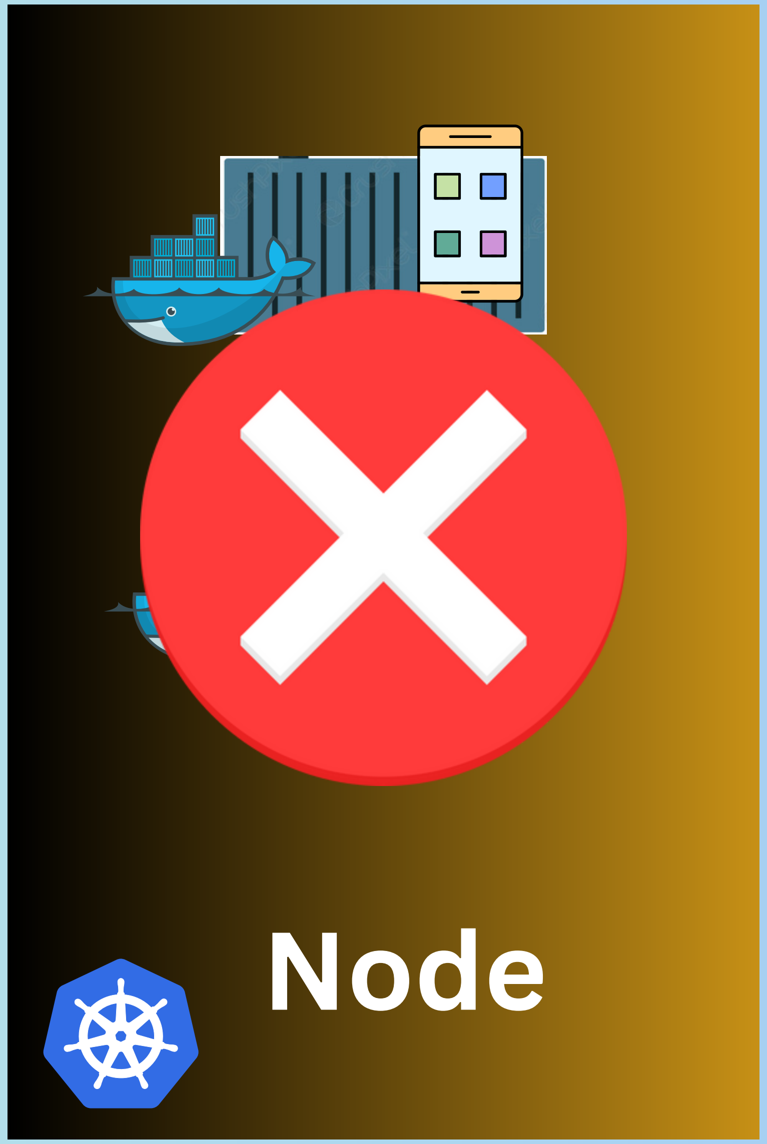


Heavy load on app

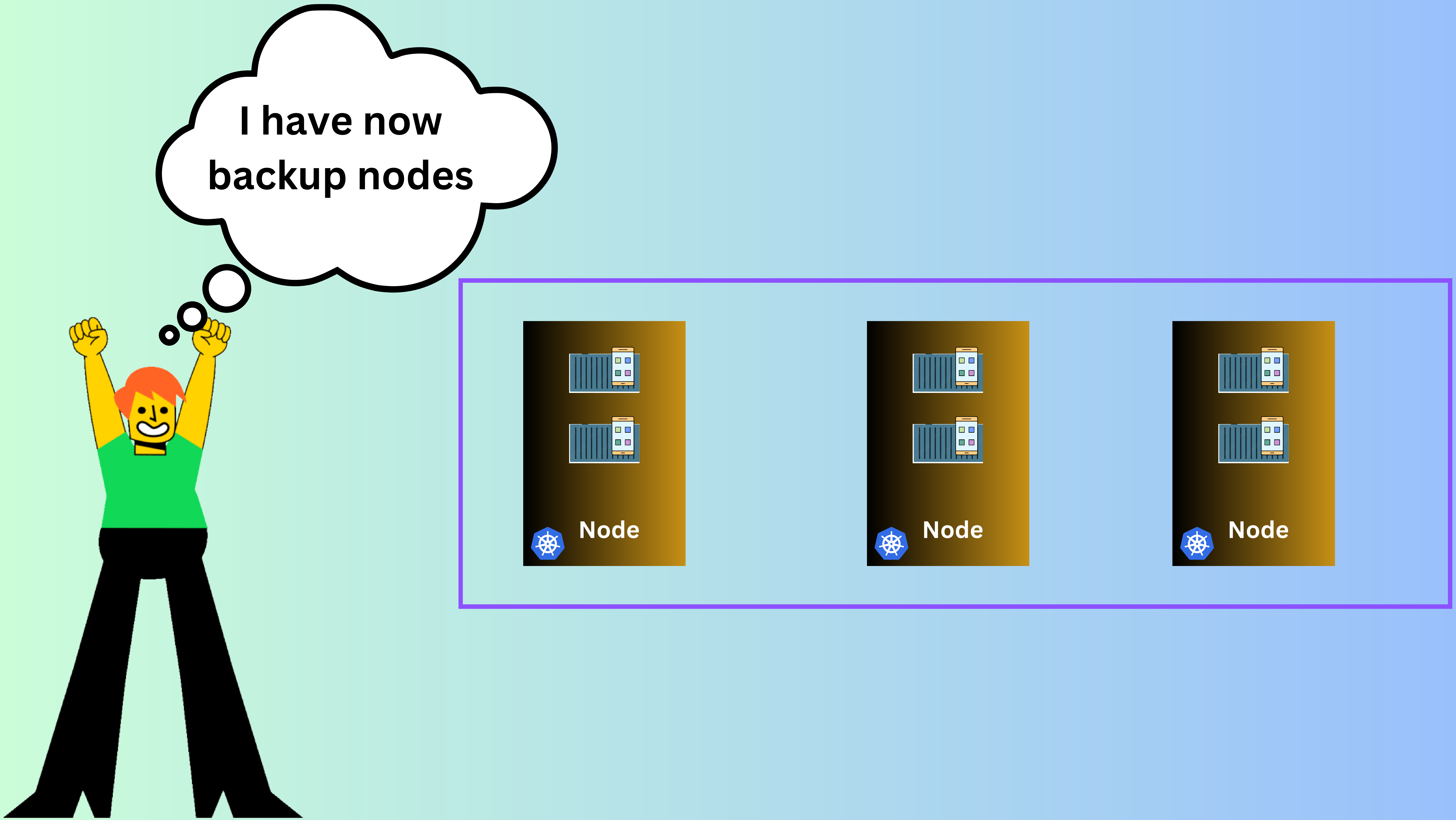


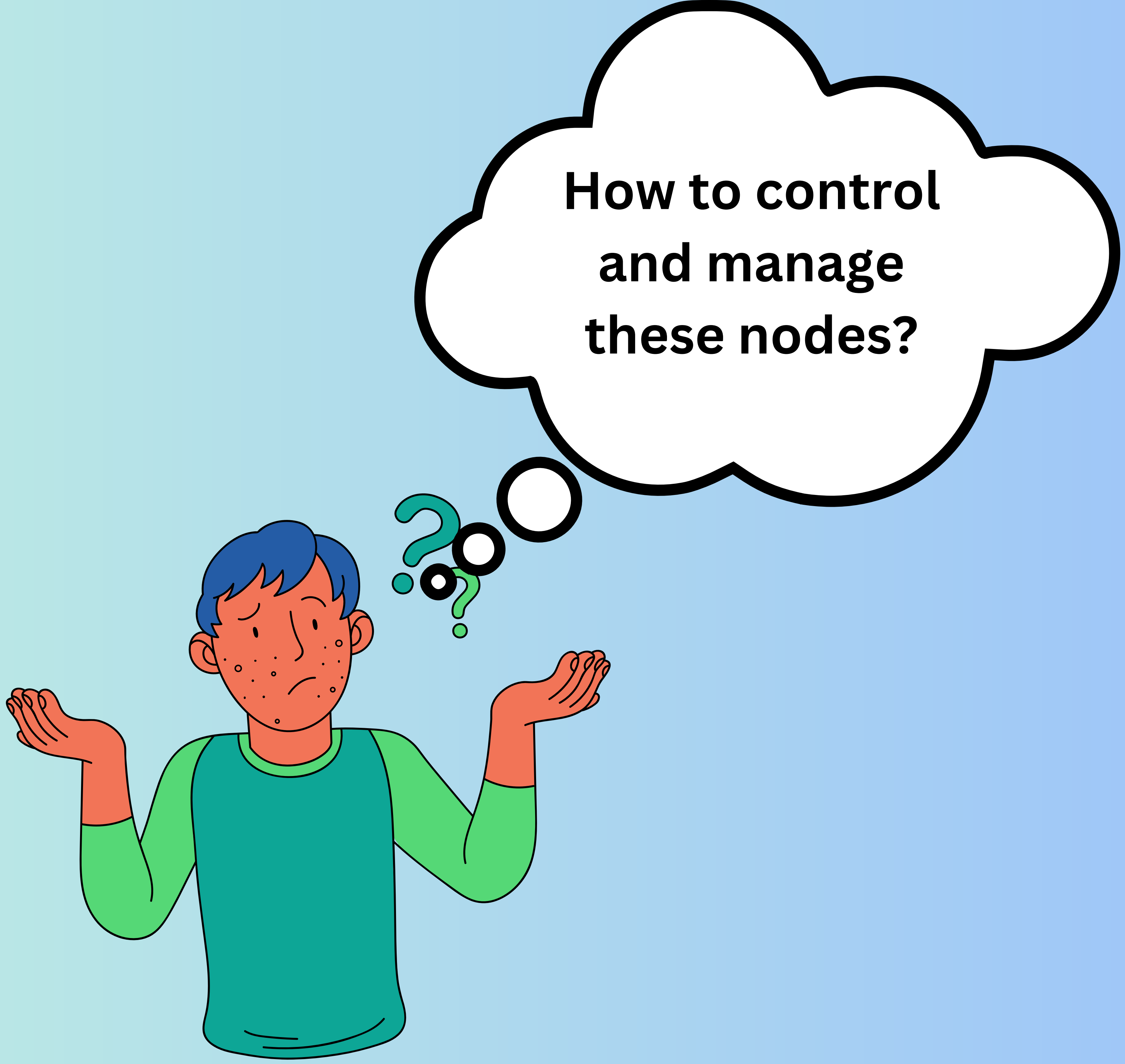
Node


What if this server fails?


Node

What should I do?


I have now backup nodes
Node
Node
Node



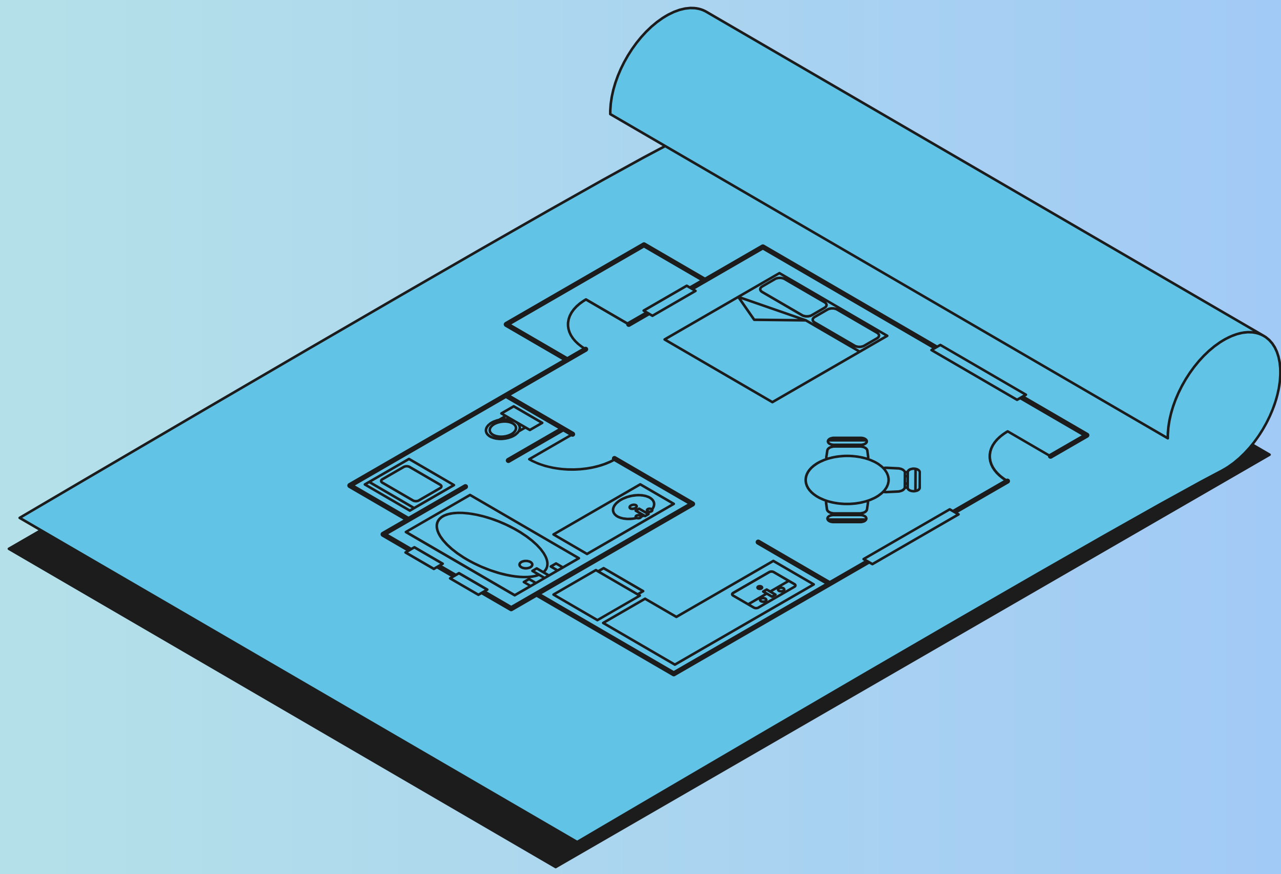
How to control and manage these nodes?

We have...

Kubernetes

Kubernetes

Architecture



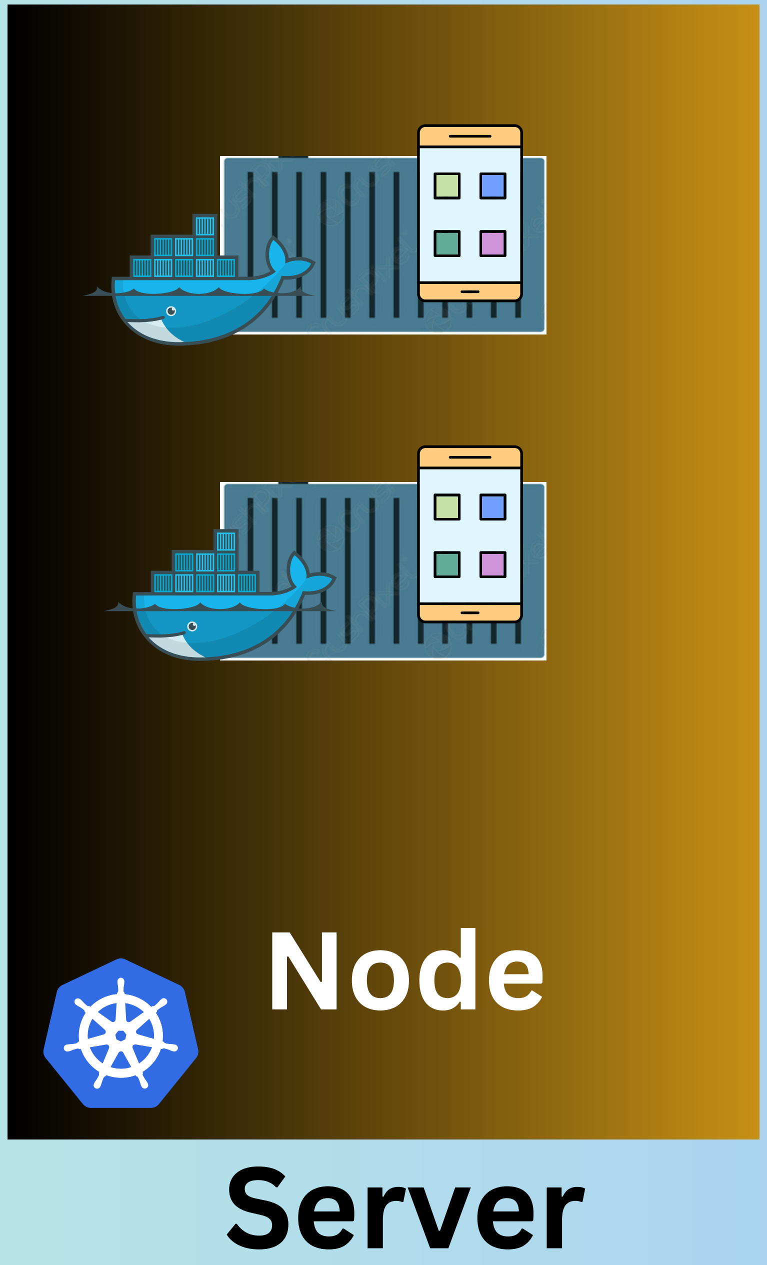
Kubernetes

When you deploy Kubernetes, you get a cluster.

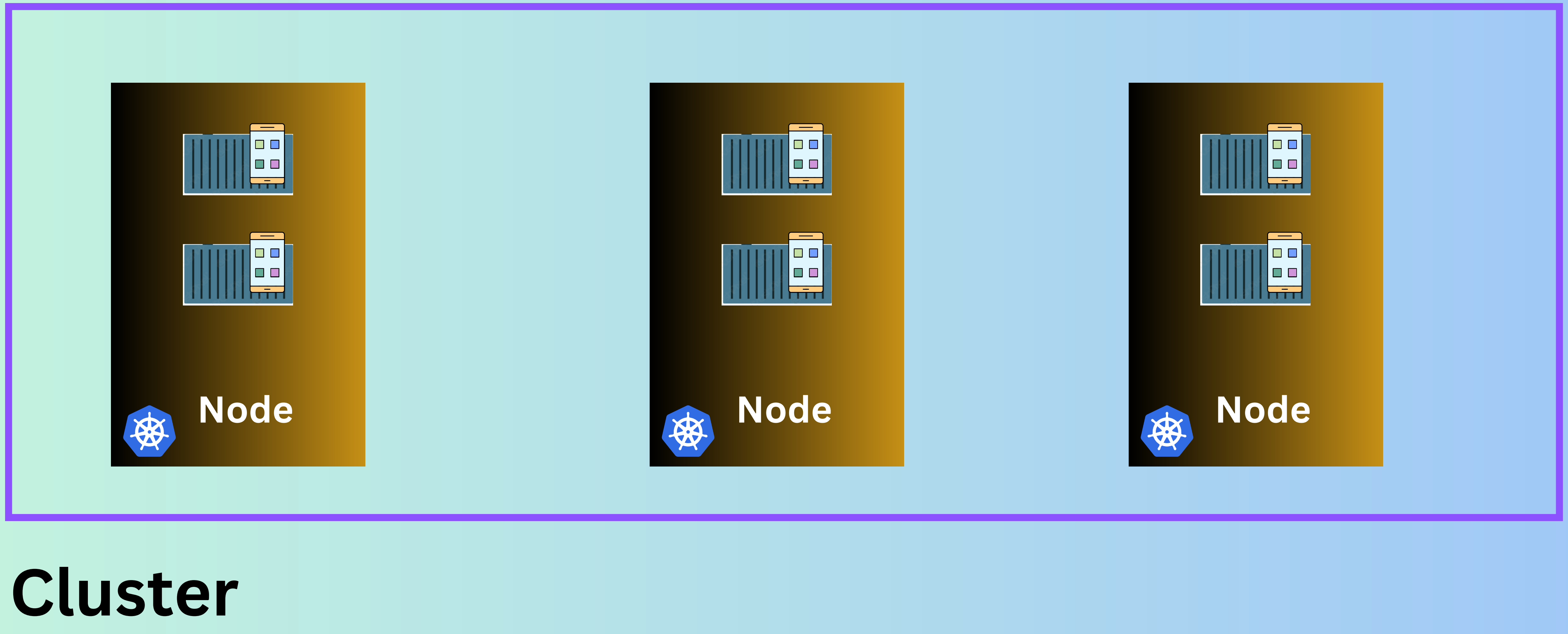
Two important parts are:

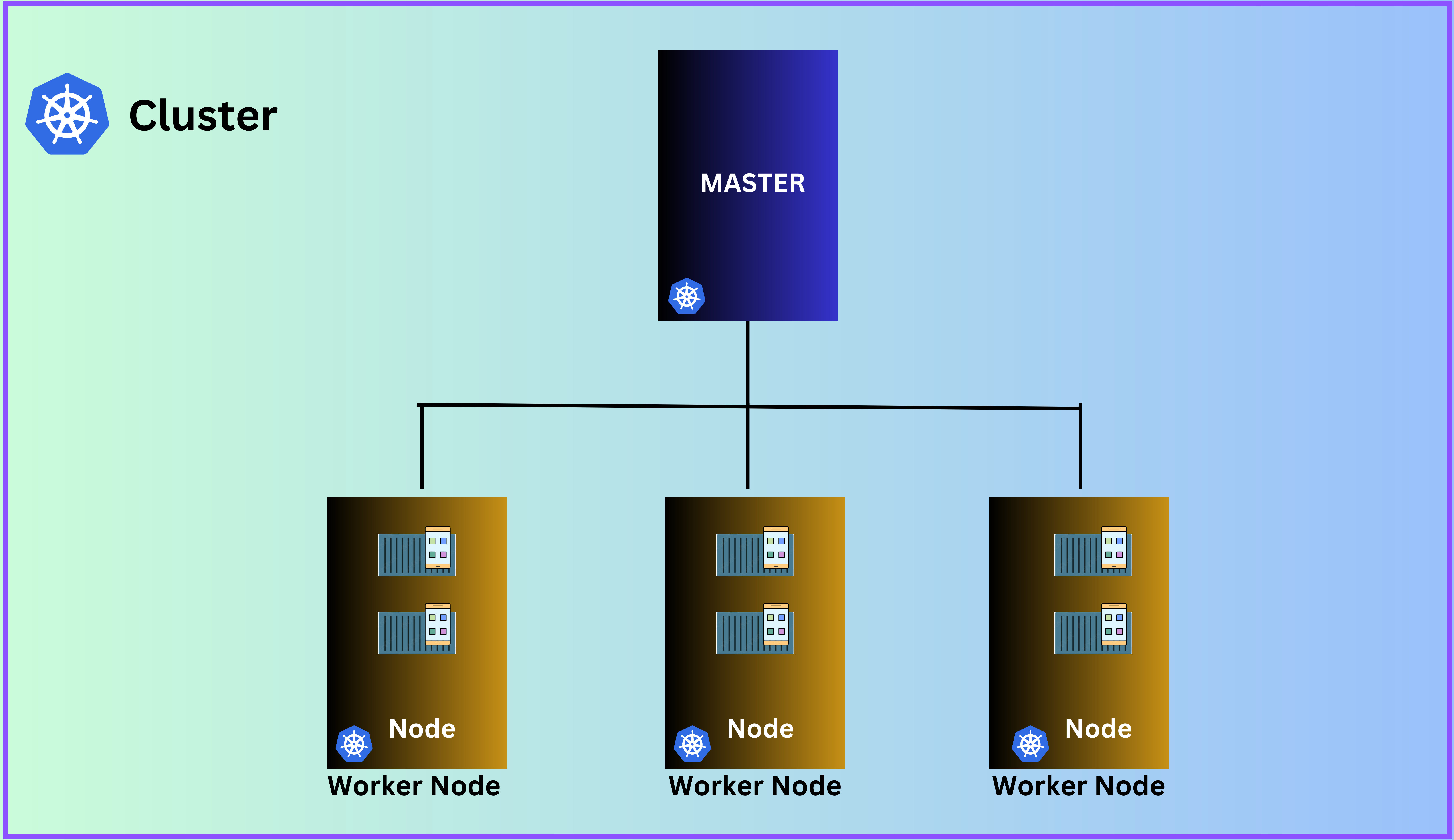
- Master (Control Plane) &
- Worker nodes.

Nodes (Minions)



We need a way to manage these nodes...





Cluster

MASTER

Node

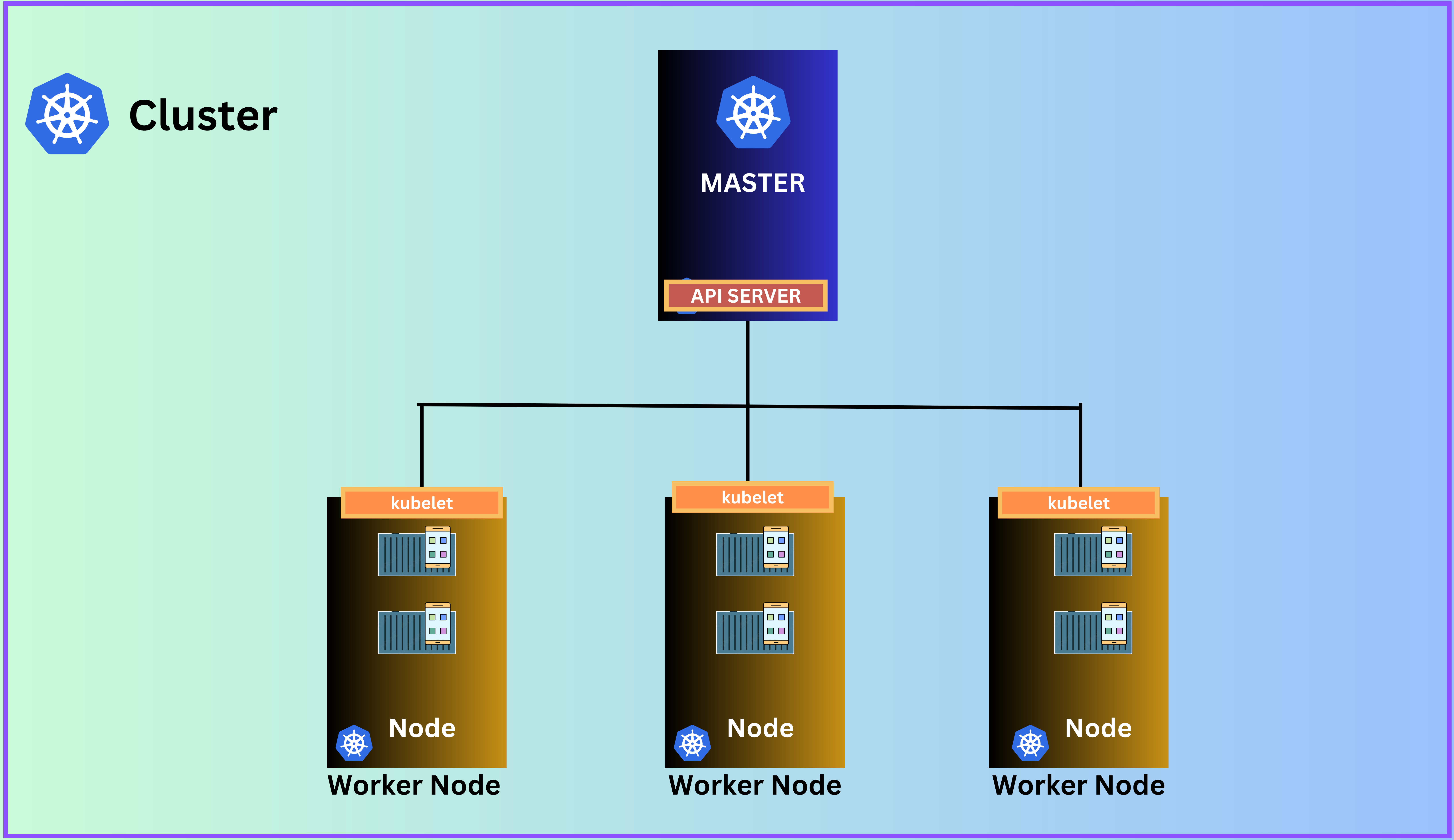
Worker Node

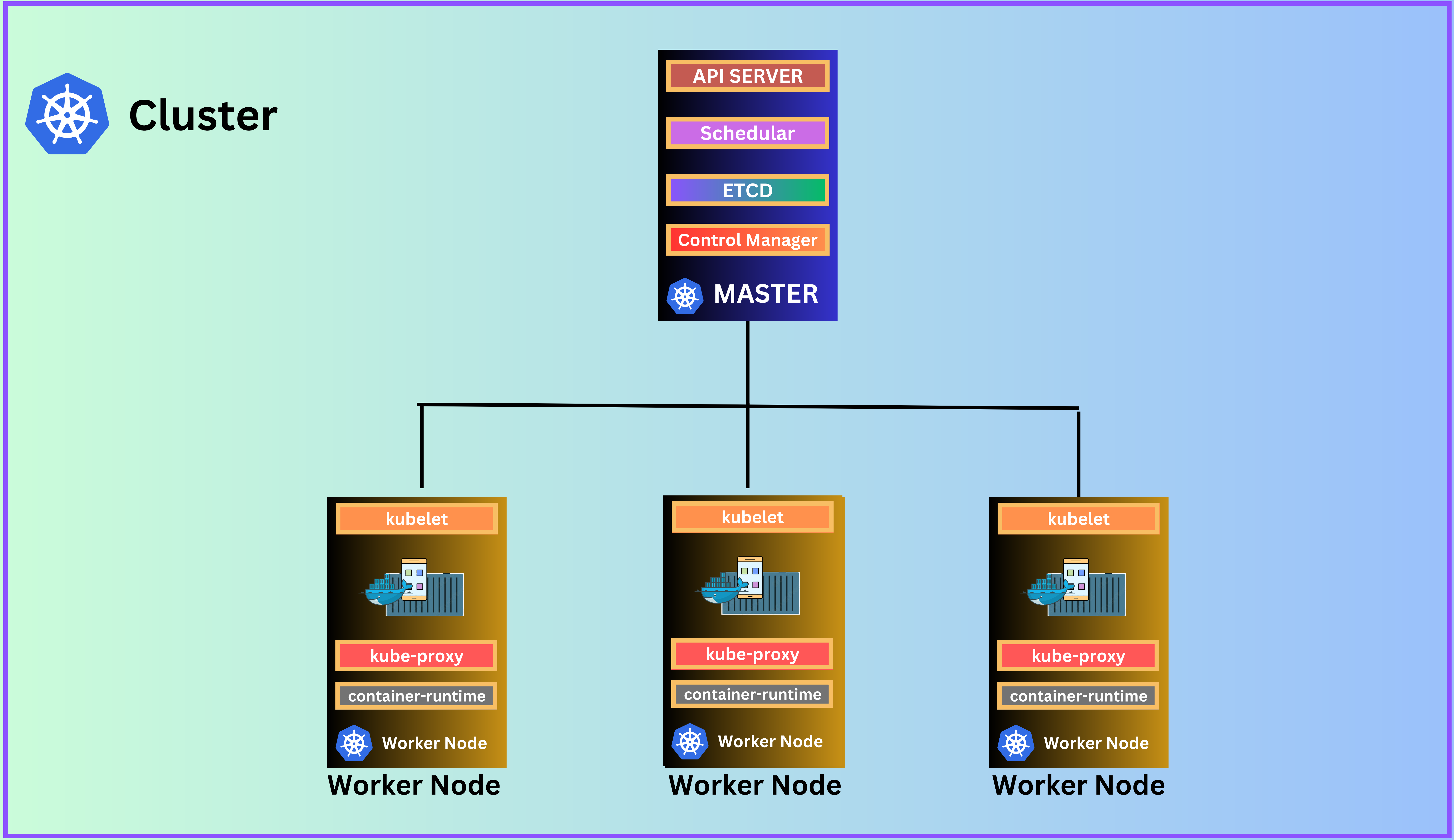
Node

Worker Node

Node

Worker Node


Cluster
MASTER
API SERVER
kubelet
kubelet
kubelet
Node
Node
Node
Worker Node
Worker Node
Worker Node



Cluster

API SERVER

Schedular

ETCD

Control Manager

MASTER

kubelet

kube-proxy

container-runtime

Worker Node

Worker Node

kubelet

kube-proxy

container-runtime

Worker Node

Worker Node

kubelet

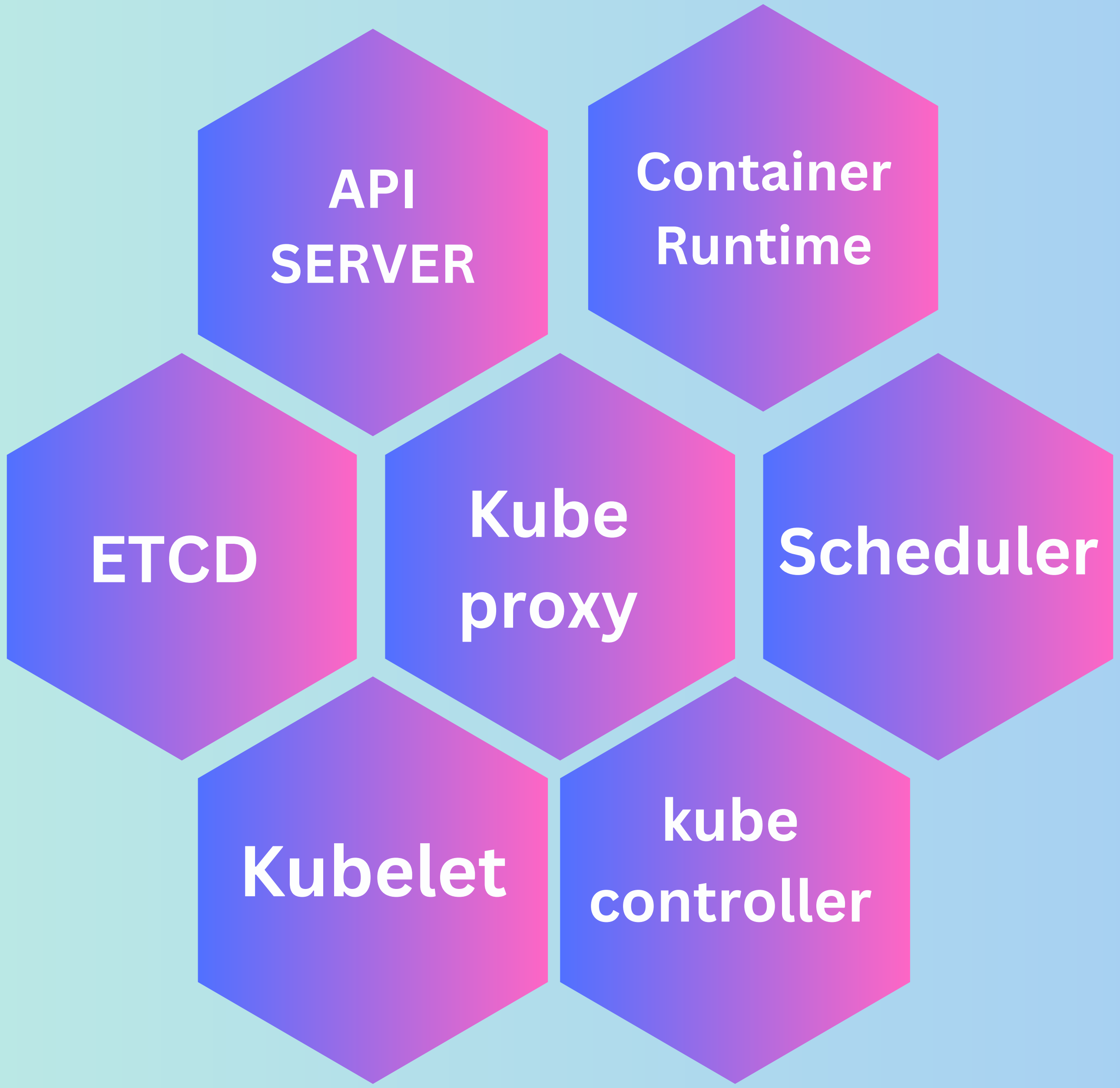
kube-proxy

container-runtime

Worker Node

Worker Node

Components



API SERVER

Container Runtime

ETCD

Kube proxy

Scheduler

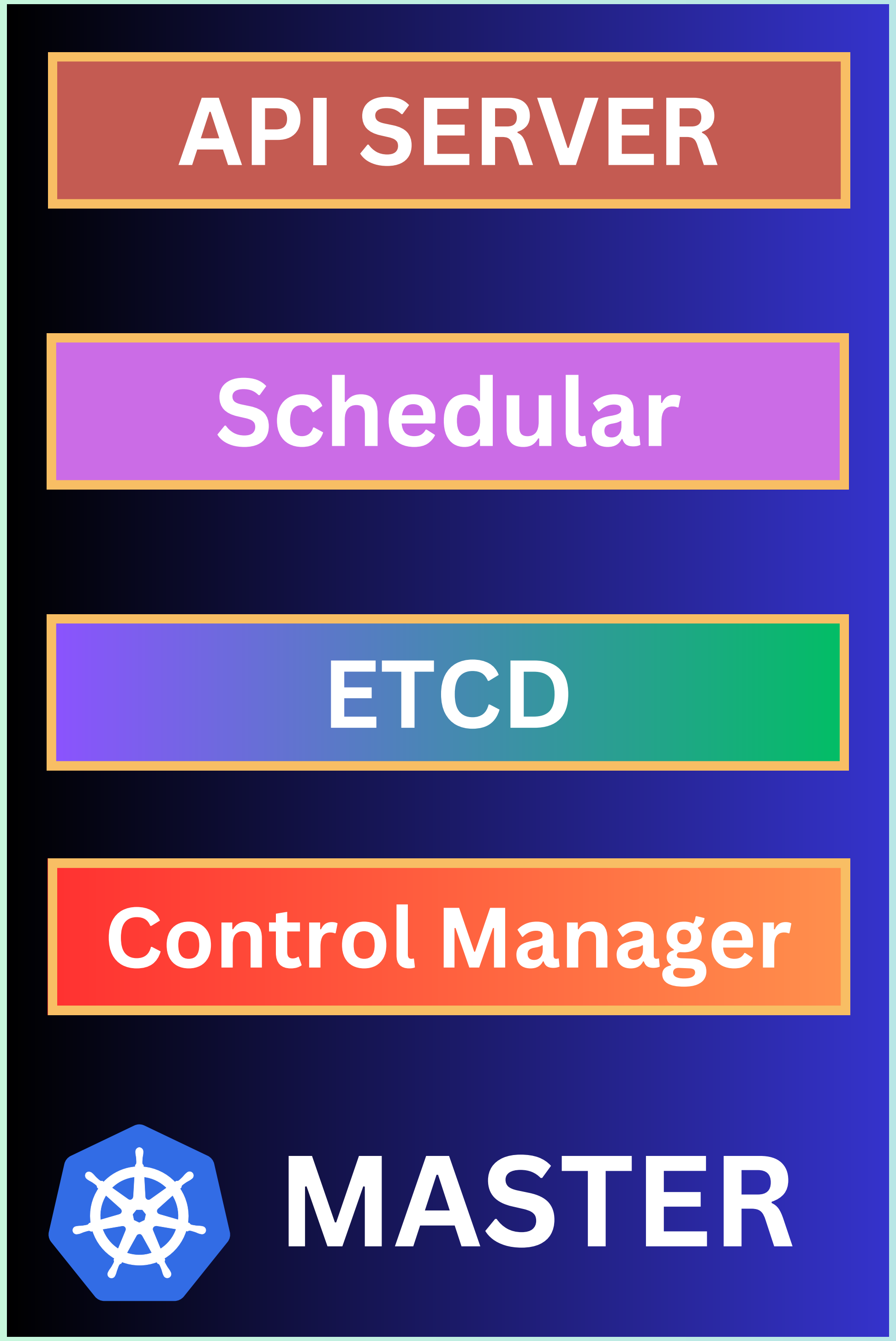
Kubelet

kube controller

What is a pod?

A single instance of a running process in a cluster.

It can run one or more containers and share the same resources.



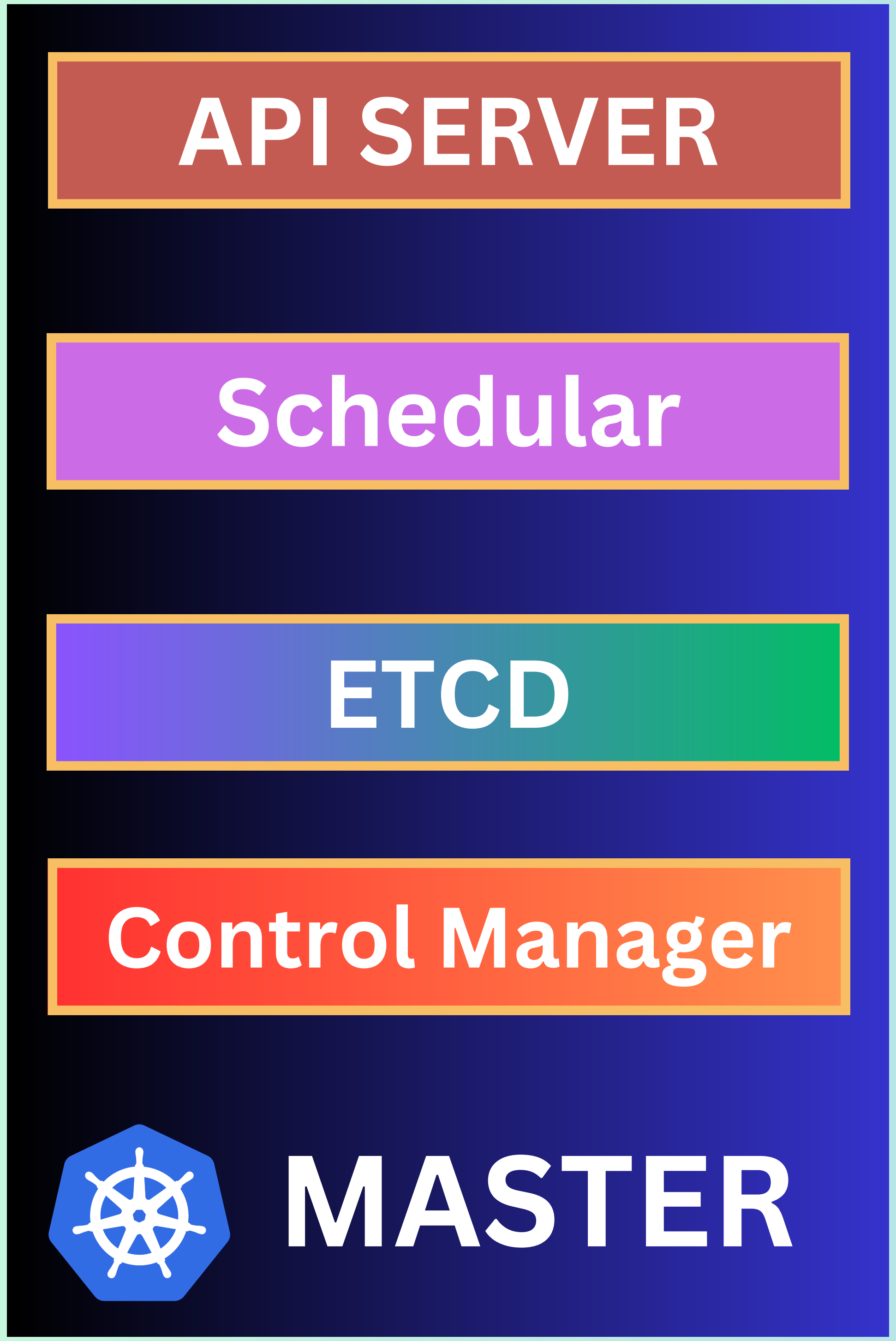
API SERVER

Schedular

ETCD

Control Manager

MASTER



API SERVER

Schedular

ETCD

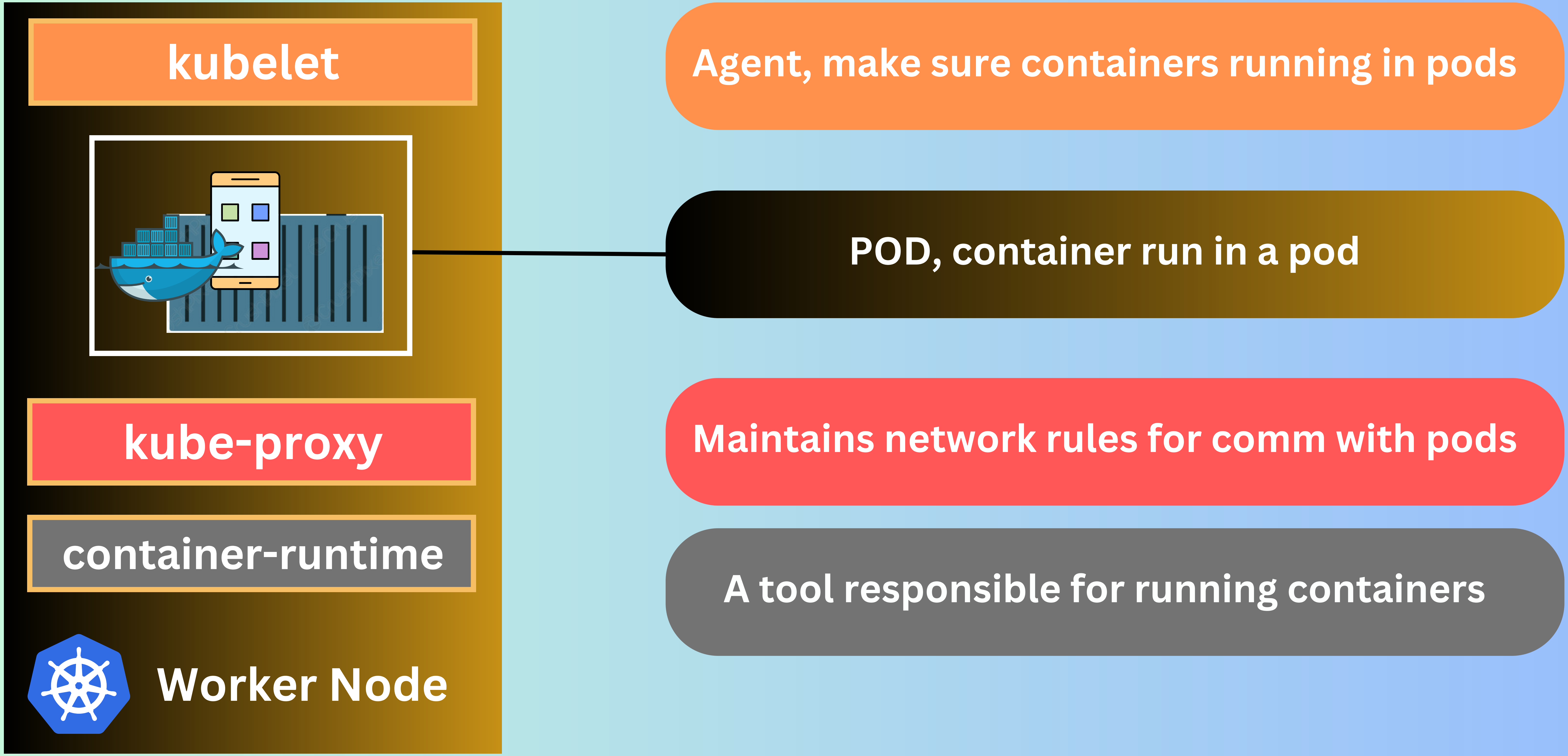
Control Manager

MASTER

assign node to newly created Pods

key-value store, having all cluster data

responsible for managing the state of the cluster



kubelet

Agent, make sure containers running in pods

POD, container run in a pod

kube-proxy

Maintains network rules for comm with pods

container-runtime

A tool responsible for running containers

Worker Node

With Kubernetes

Container Orchestration

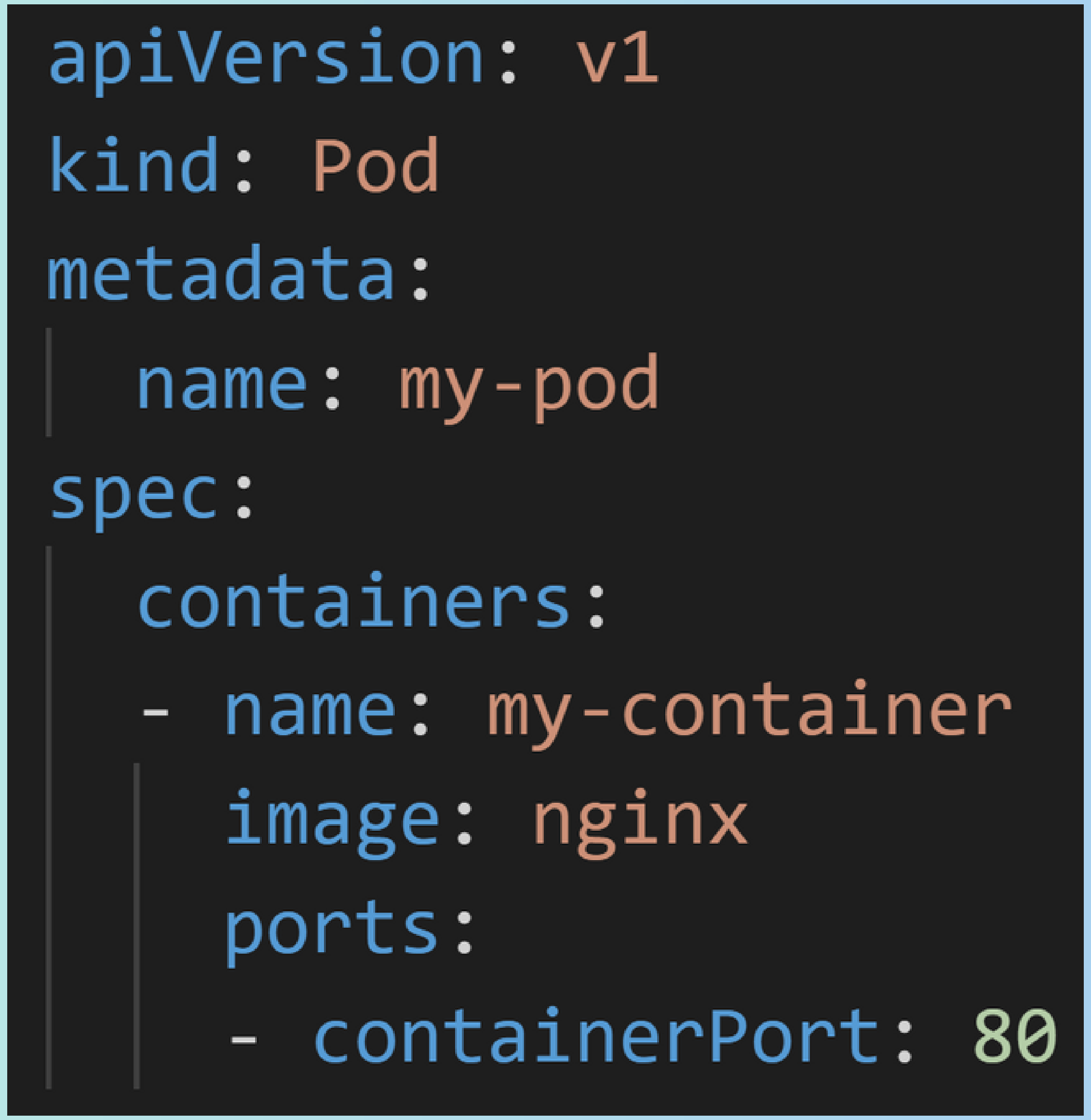
Scalability

Load Balancing

High Availability

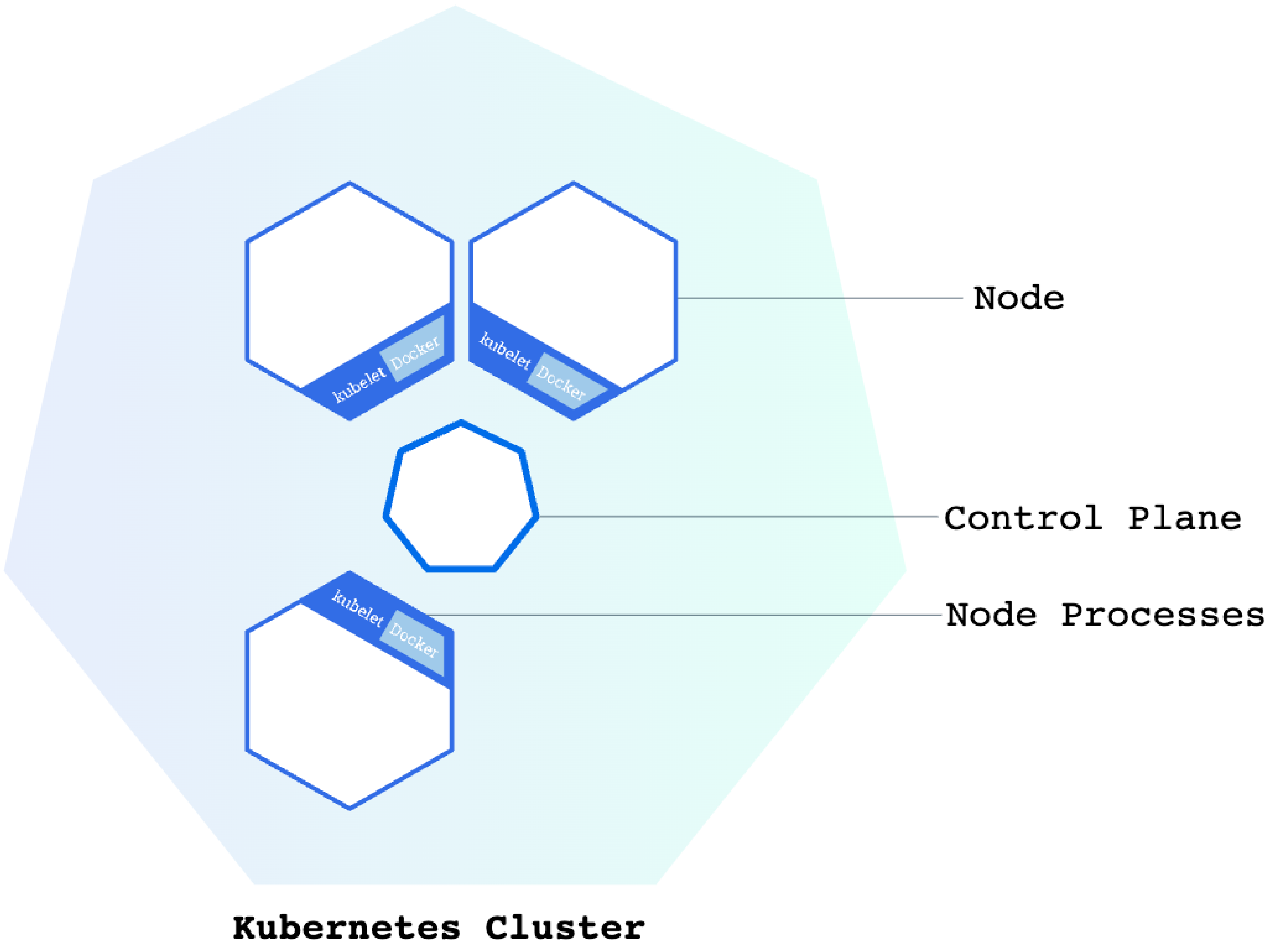
Rollouts & Rollback

Sample yaml Config file,,



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-container
    image: nginx
    ports:
    - containerPort: 80
```

Cluster Diagram

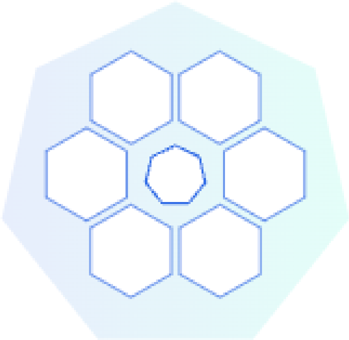


- **minikube start/delete**
- **minikube status**
- **minikube dashboard**

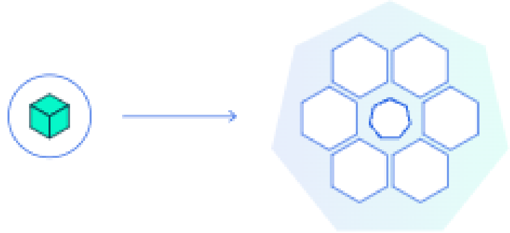
- **kubectl create deployment my-app --image=link**
- **kubectl get deployments**
- **kubectl get pods**
- **kubectl delete deployment my-app**

- **kubectl expose deployment my-app --type=LoadBalancer --port=80**
- **minikube service my-app**
- **kubectl get services**

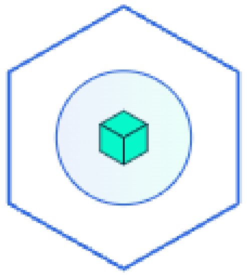
Kubernetes Basics Modules



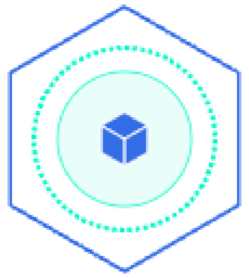
1. Create a Kubernetes cluster



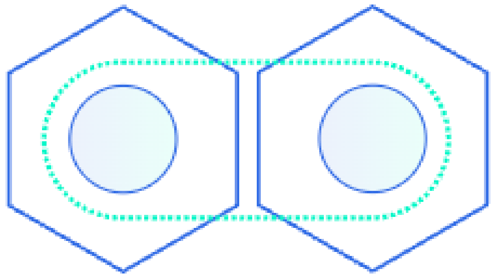
2. Deploy an app



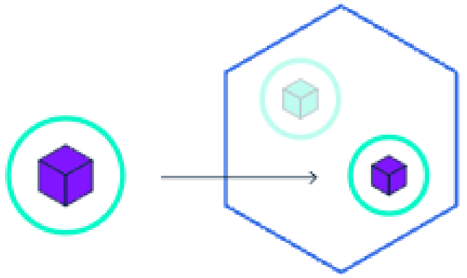
3. Explore your app



4. Expose your app publicly

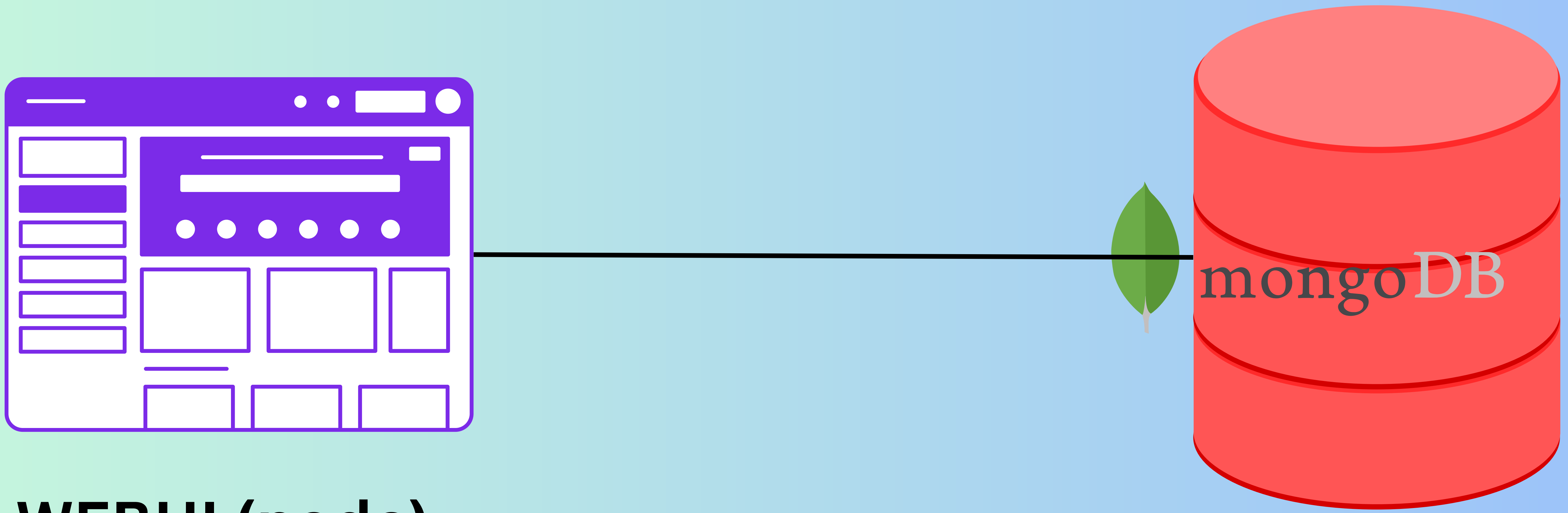


5. Scale up your app



6. Update your app

Multi Container Application



WEBUI (node)

mongodb

```
docker pull philippaul/node-mongo-db:01
```

To Run MongoDb

```
docker run -d -p 27017:27017 --network my-net --name mongo mongo
```

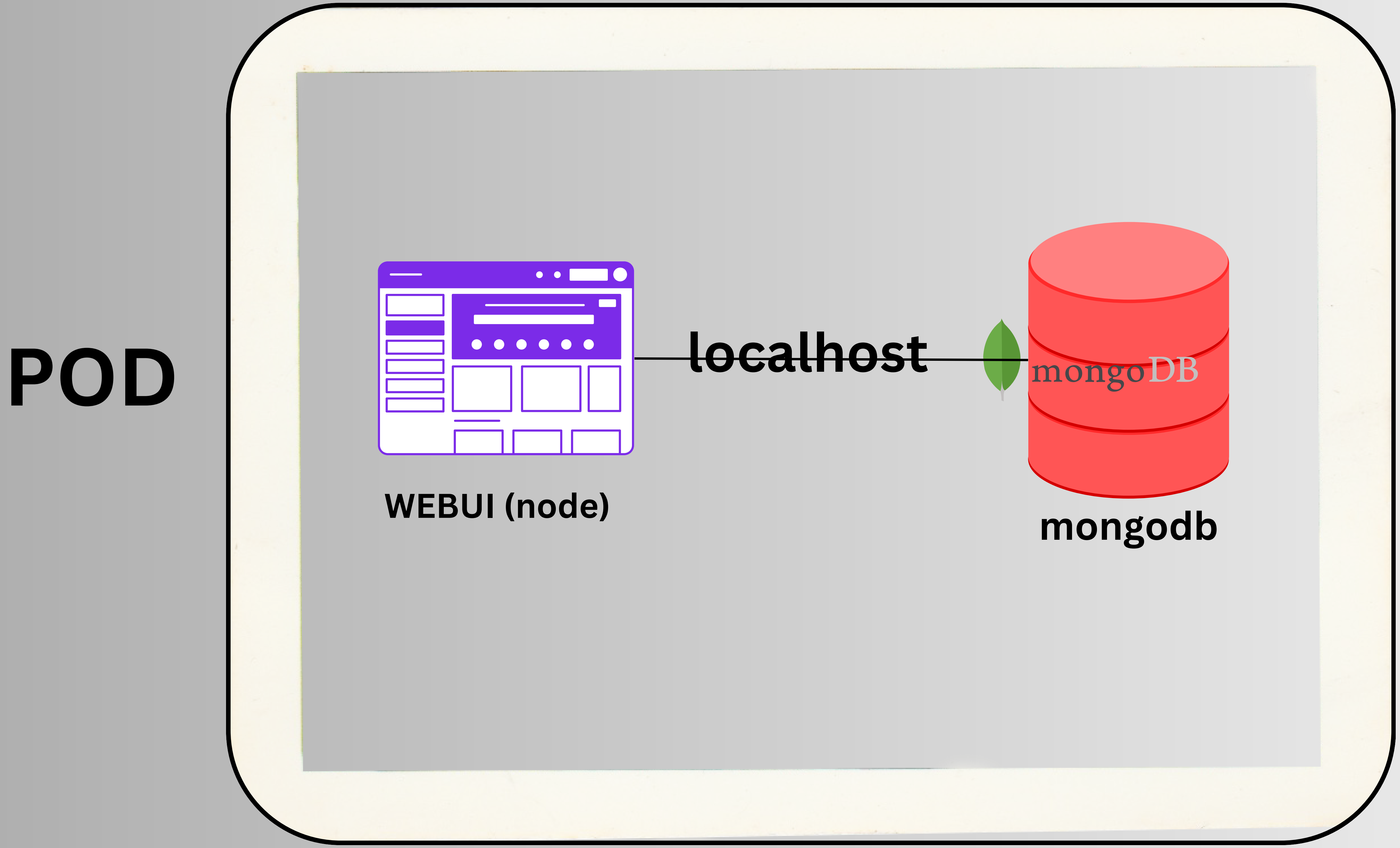
To Run our Node App

```
docker run --network my-net -p 3000:3000 --name myapp philippaul/node-mongo-db:01
```

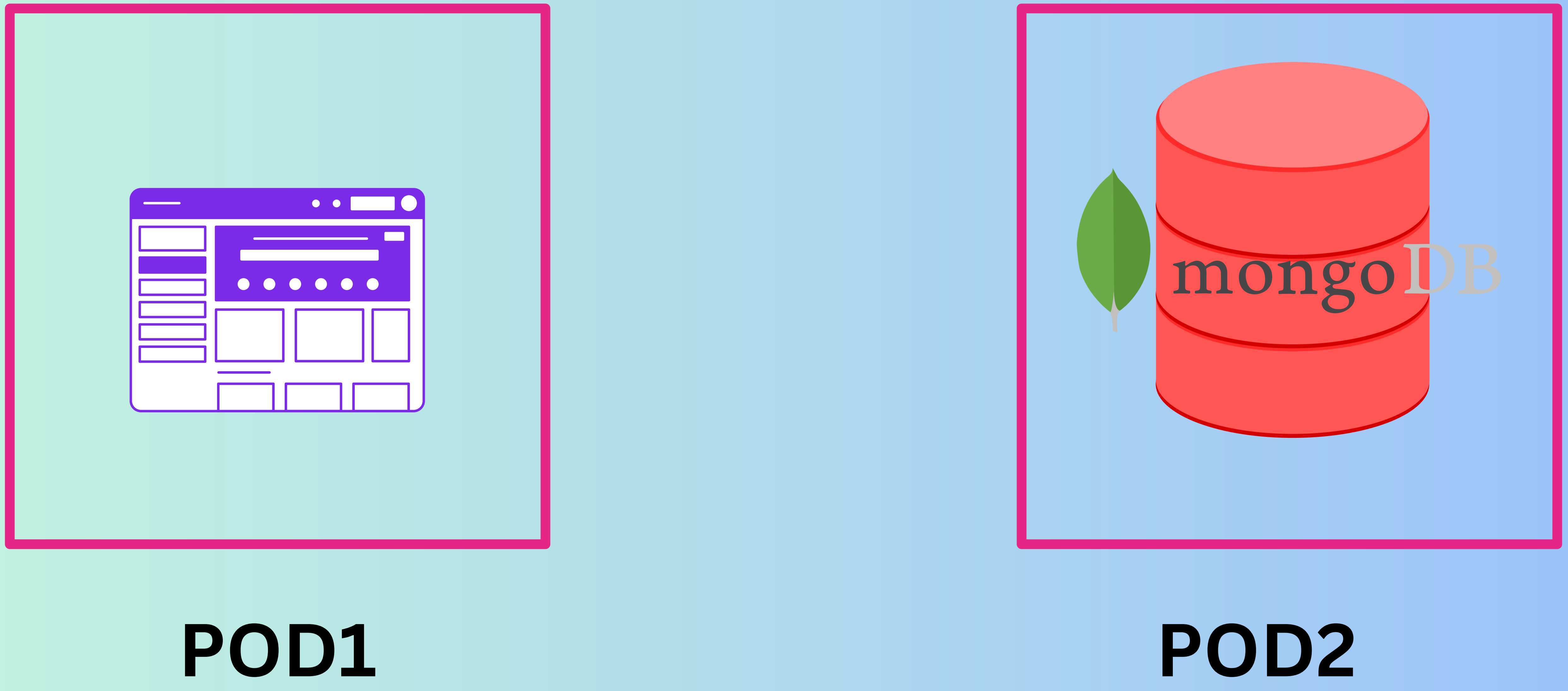
There are two ways to run application with multiple containers -

- Run multiple containers in same pod
- Run each container in separate pod

- Run multiple containers in same pod



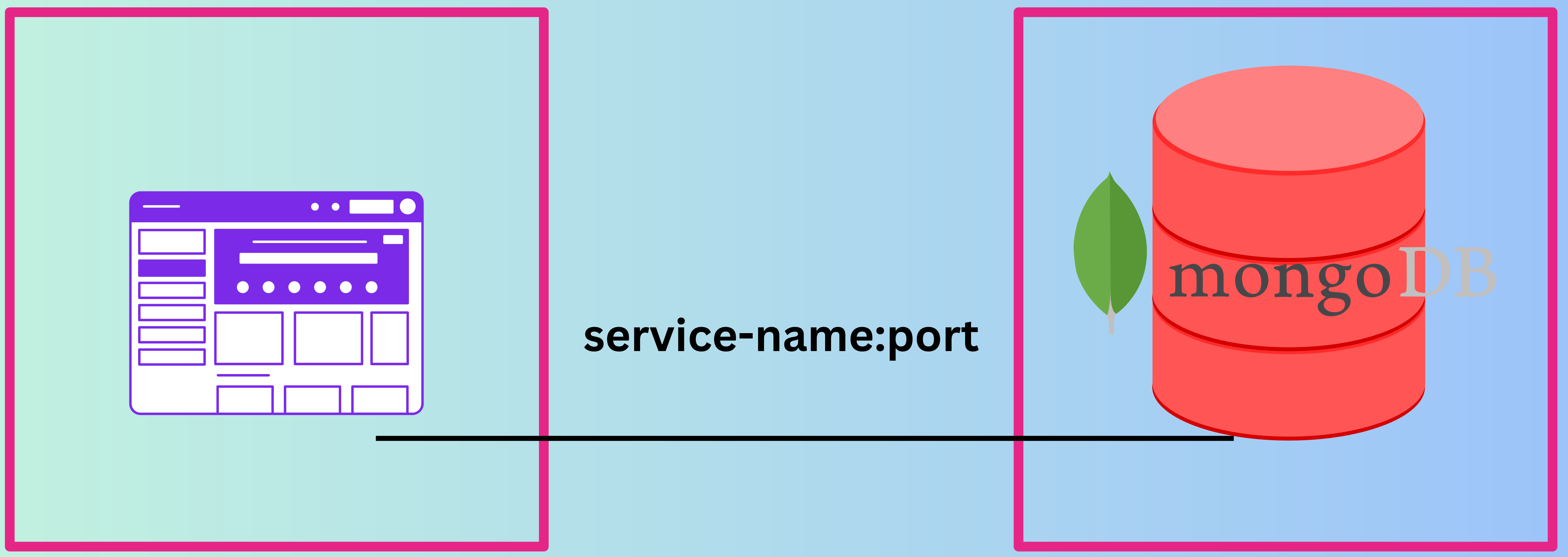
Run each container in separate pod



POD1

POD2

Run each container in separate pod



service-name:port

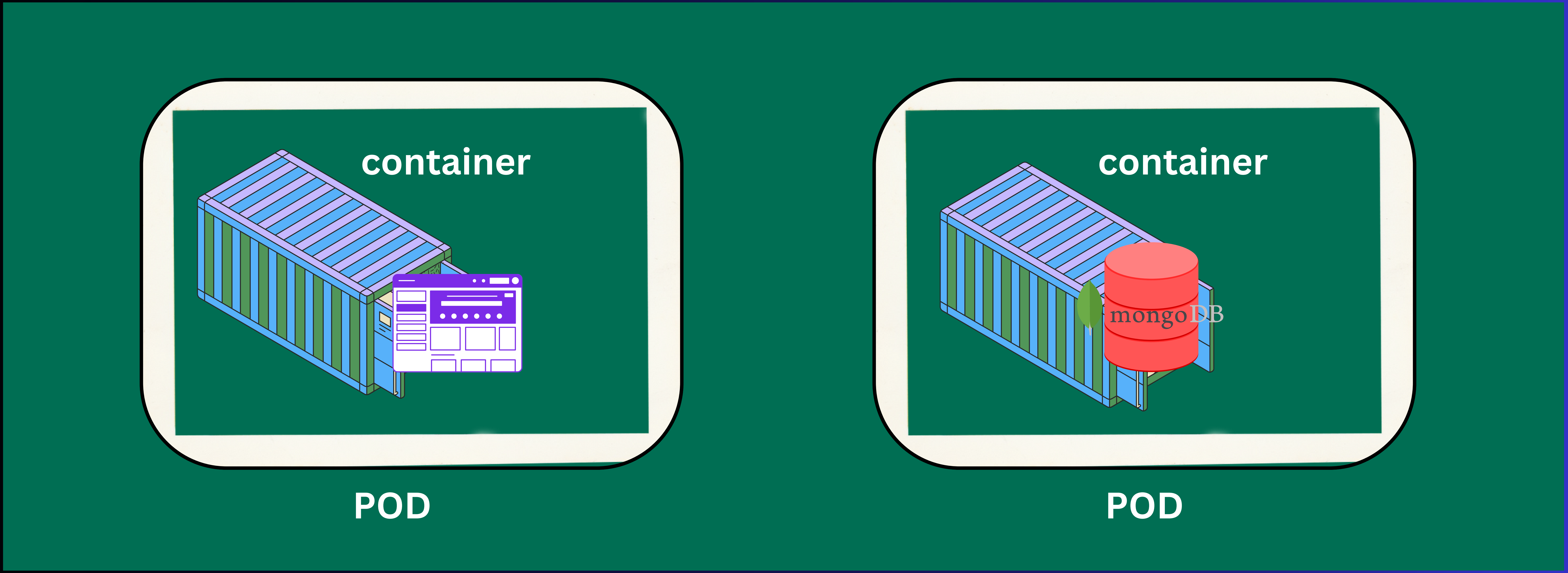
POD1

POD2

Steps:

- Changes in node project (dynamic host and port for database URL)
- Building image & pushing image with 03 version
- YAML config requirement explanation
- First create depl and service config for MONGO APP so we will be having service name
- Now create depl and service config for NODE APP and explain how to use env variables in config
- Now we will create configMAP file
- Run node-app deployment

Volumes & Data



container

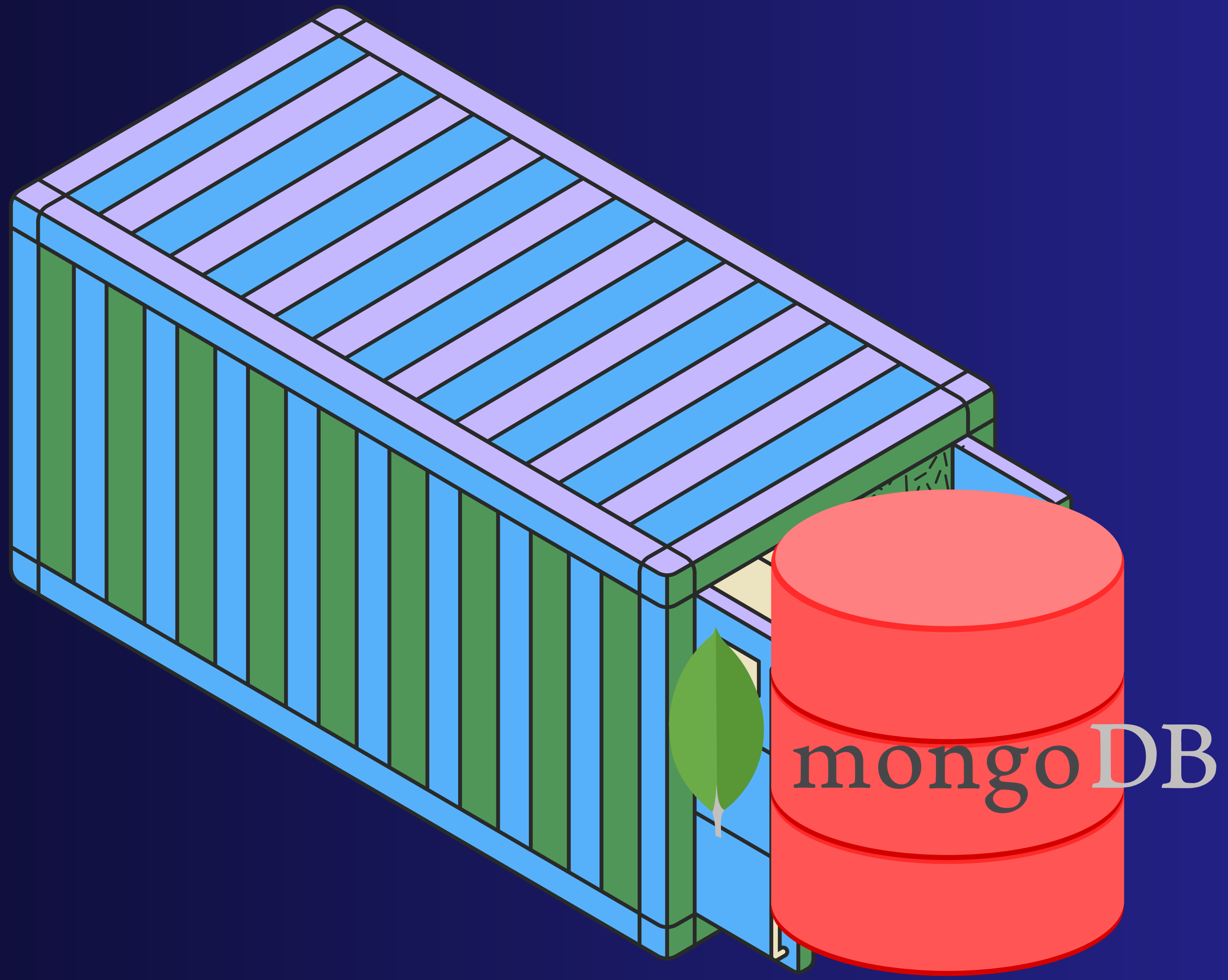
container

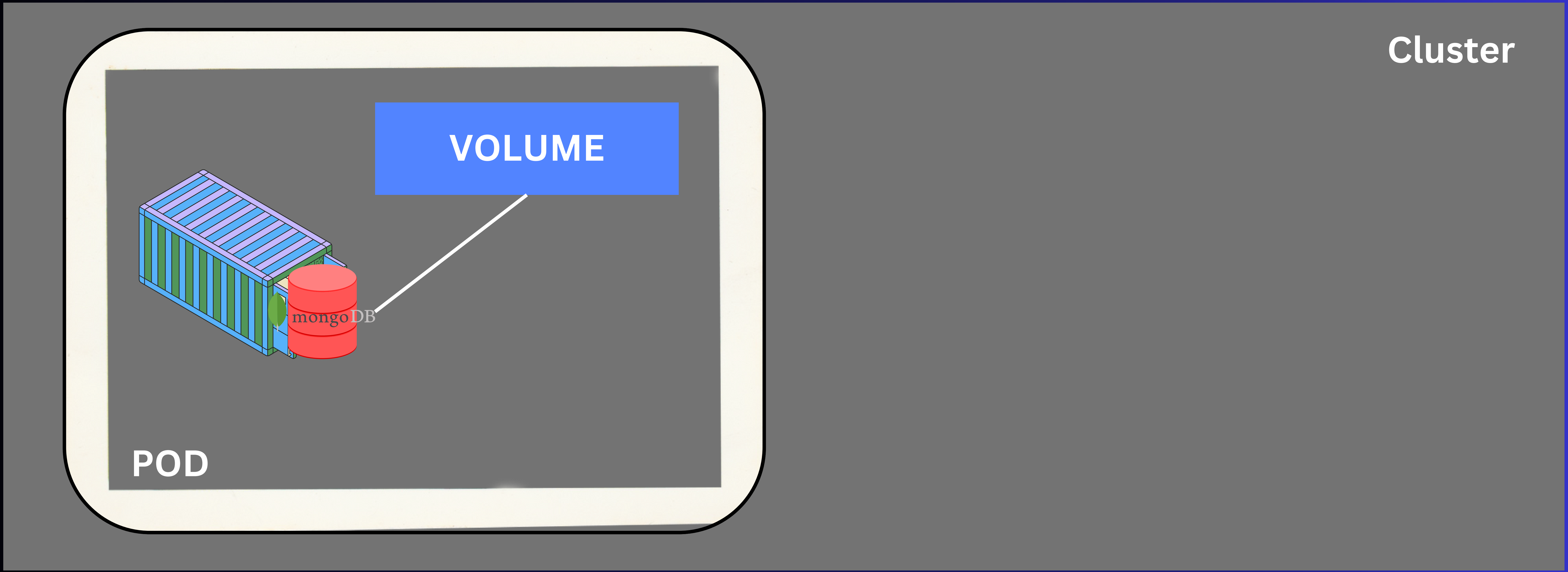
POD

POD

Cluster

Container

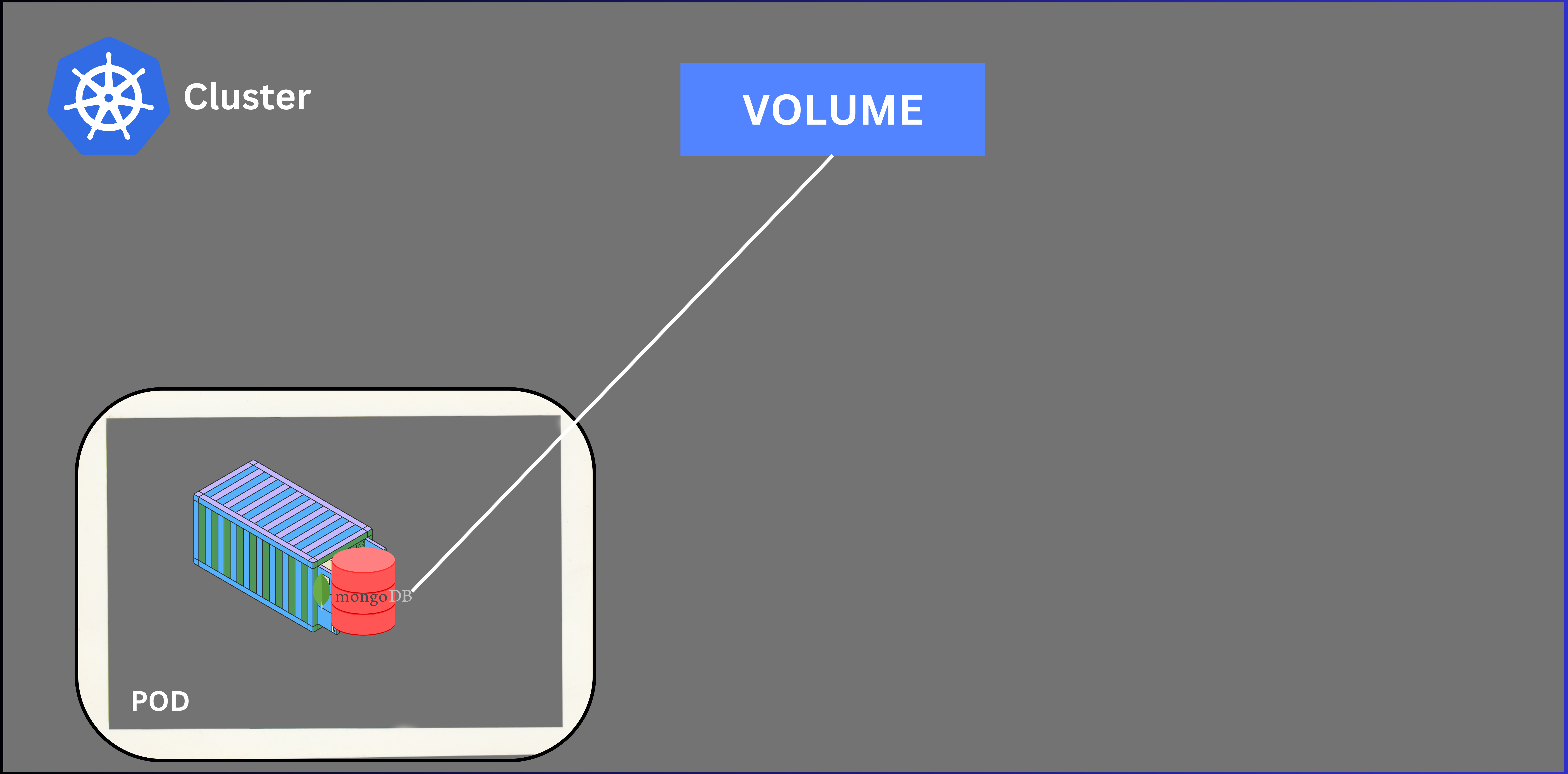




Cluster

VOLUME

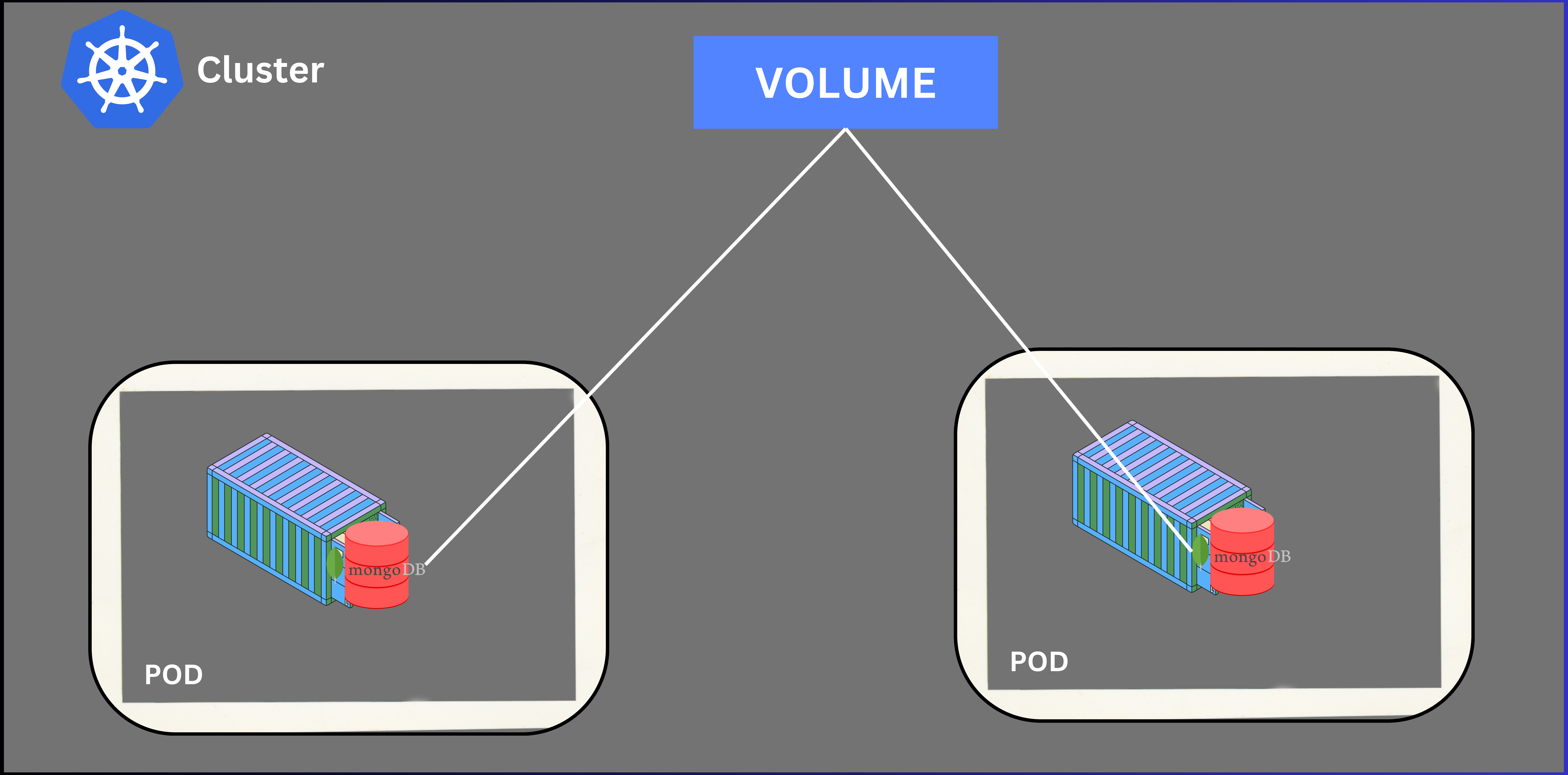
POD

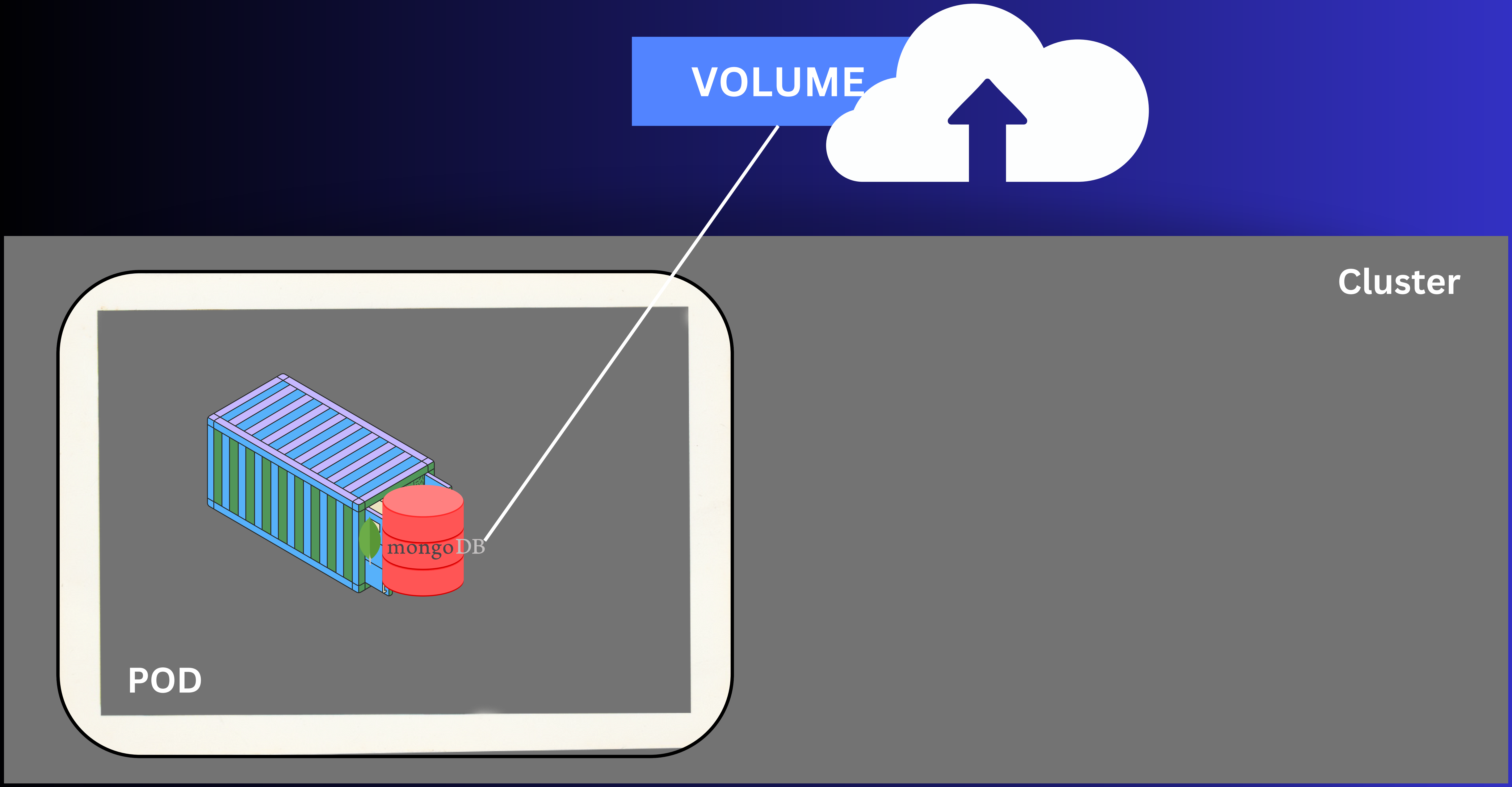


Cluster

VOLUME

POD

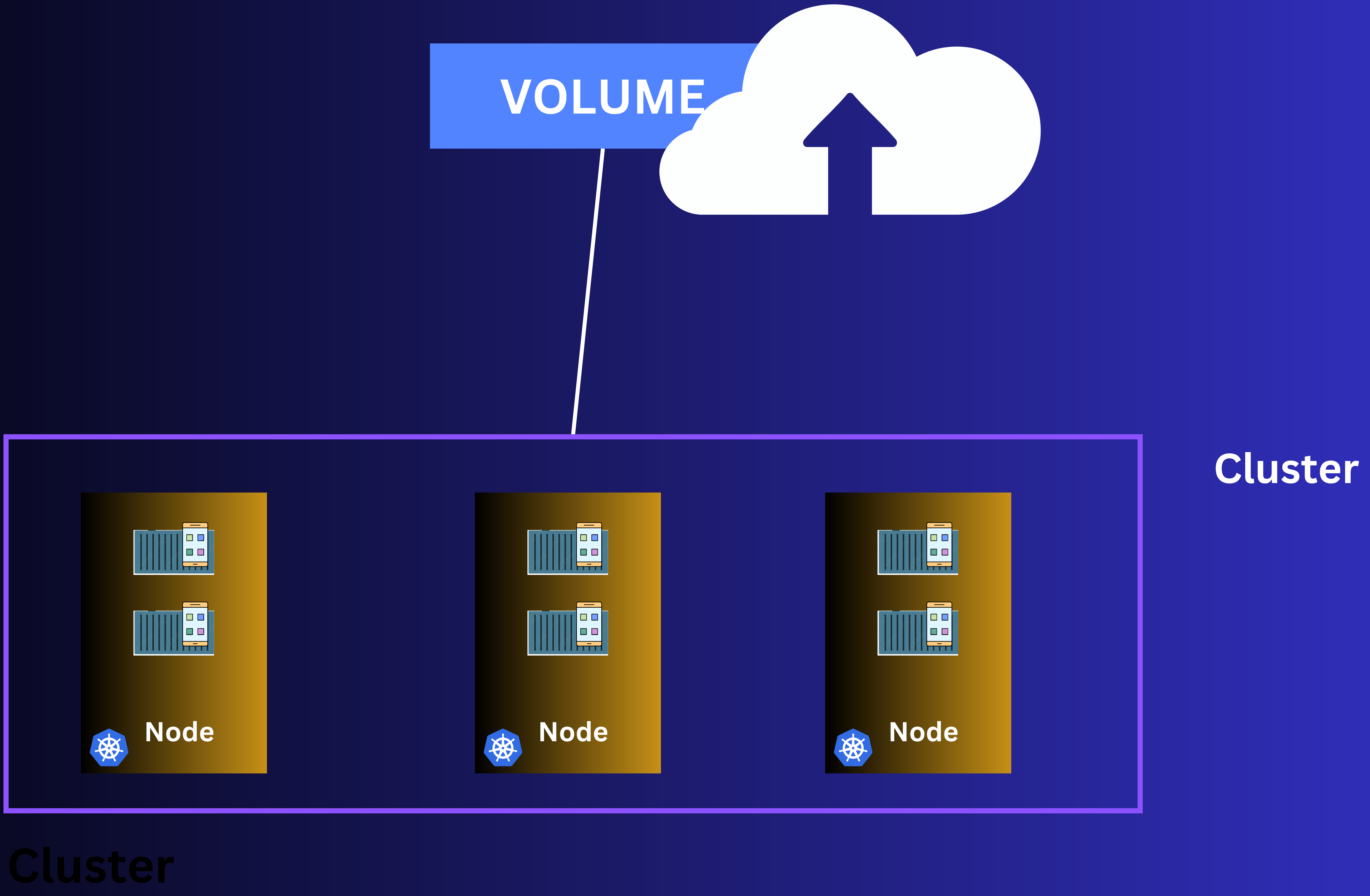

Cluster
VOLUME
mongoDB
POD
mongoDB
POD



VOLUME

Cluster

POD



VOLUME

Cluster

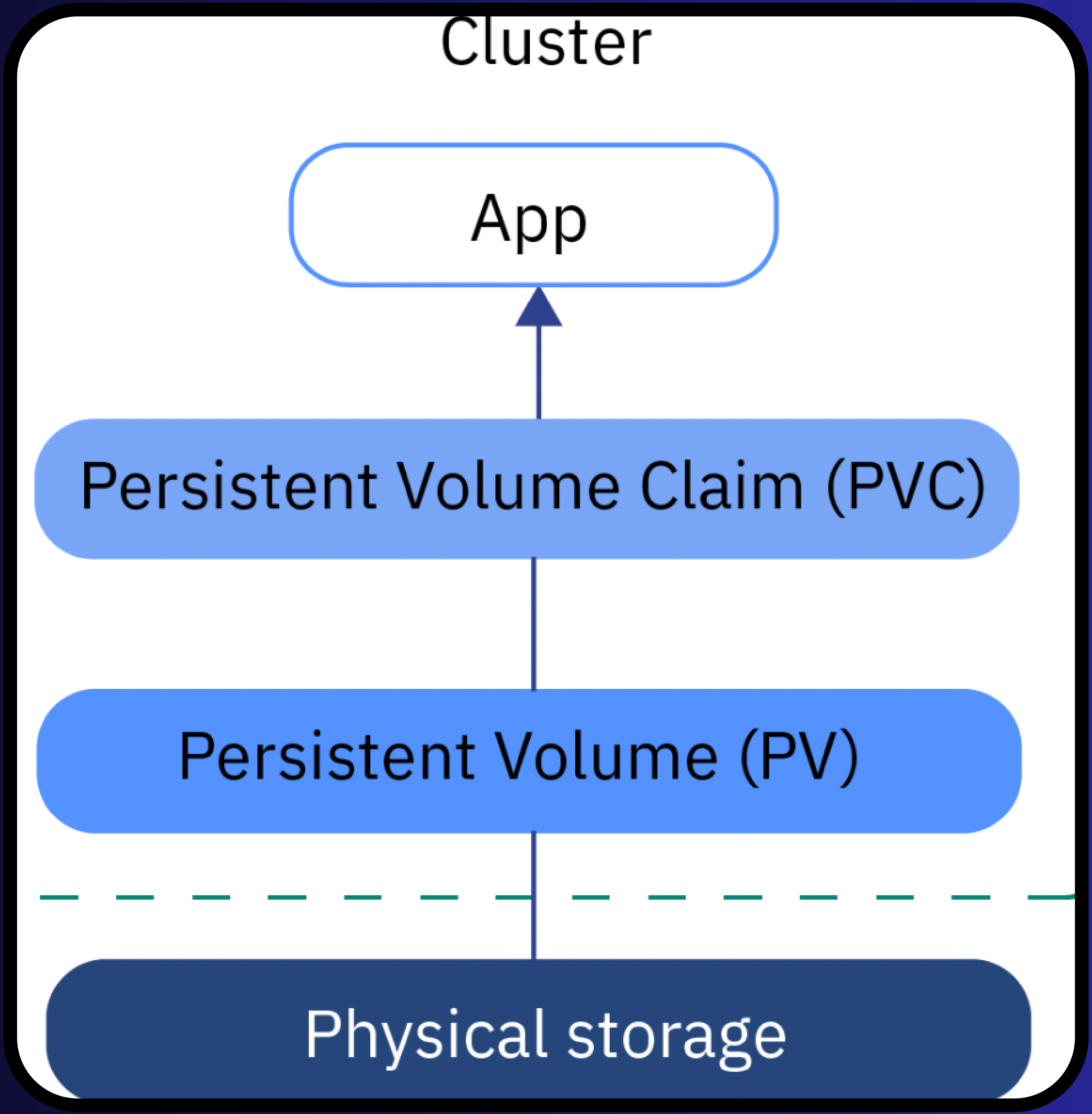
Node

Node

Node

Cluster

Persistent Volume



Cluster

App

Persistent Volume Claim (PVC)

Persistent Volume (PV)

Physical storage